Lubrication of Piston Rings in Large 2–and 4–stroke Diesel Engines

Felter, Christian Lotz

Publication date:
2007

Document Version
Publisher's PDF, also known as Version of record

Technical University of Denmark
MAN Diesel A/S
March, 2007

Lubrication of Piston Rings in Large 2– and 4–stroke Diesel Engines

Christian Lotz Felter

Advisors
Peder Klit, Department of Mechanical Engineering,
Technical University of Denmark
Anders Vølund, New Design, MAN Diesel A/S

Preface

This thesis is submitted as partial fulfillment of the requirements for the Industrial PhD-degree in Denmark. The work is done in collaboration with MAN Diesel A/S, Copenhagen. The work has been carried out from January 2004 to March 2007 at the Department of Mechanical Engineering, Technical University of Denmark, Kgs. Lyngby. The project has been supervised by Associate Professor PhD Peder Klit and Senior Research Engineer PhD Anders Vølund.

Christian Lotz Felter

Lubrication of Piston Rings in Large 2– and 4–stroke Diesel Engines

Abstract

Piston rings are vital components of any internal combustion engine, and their performance affect important properties such as frictional losses, oil consumption, and wear of parts. This thesis deals with the lubrication of piston rings from a theoretical point of view. Predictions are made using numerical models implemented as computer programs.

The classical Reynolds equation can be used to calculate the pressure distribution in thin films of fluid. In relation to piston ring lubrication it is, however, less straight forward to apply the Reynolds equation since the inlet (and outlet) point of the lubricated conjunction is not known before hand. In order to overcome this problem, which is the main topic of this work, two different paths are followed. First an equation for the inlet point location is derived under the assumption of steady-state running conditions. Assuming that this limitation is fulfilled in a quasi-static sense a concrete example is analyzed using the Reynolds equation. Next a free surface 2D code based on the compressible Navier–Stokes equations is developed. The main idea is to model also the oil film outside the piston ring. Through time integration the movement of the inlet point on the piston ring can be calculated.

While the model based on the Reynolds equation has a shortcoming with respect to the assumption of steady-state conditions, the model based on the Navier–Stokes equations turns out to be computationally expensive. Different remedies such as implicit time stepping or combination of the Navier–Stokes equations and the Reynolds equation are examined.

The text includes detailed derivations of the models that have been used, and the numerical properties are assessed by convergence studies and comparison with benchmark problems.

Smøring af stempelringe i store 2– og 4–takts dieselmotorer

Abstract (dansk)

Stempelringe udgør en vital komponent i enhver forbrændingsmotor, og deres ydeevne har betydning for vigtige egenskaber så som friktionstab, olieforbrug og slid. Denne afhandling behandler smøring af stempelringe fra et teoretisk udgangspunkt. Forudsigelserne baserer sig på numeriske modeller, der er implementeret som computerprogrammer.

Den klassiske *Reynolds ligning* kan anvendes ved beregning af trykfordelingen i en tynd væskefilm. I forbindelse med smøring af stempelringe er det imidlertid mindre oplagt hvordan ligningen skal anvendes, idet indløbs– og udløbspunkterne for smøreforbindelsen ikke er kendt på forhånd. Denne problematik, som udgør hovedemnet for dette arbejde, angribes på to forskellige måder. I den første metode udledes en ligning for indløbspunktets placering, under forudsætning af stationære kørselsforhold. Det antages at forudsætningerne gælder i en kvasi-statisk forstand, hvorefter et konkret eksempel behandles. I den anden metode udvikles et to-dimensionalt simuleringsprogram baseret på *Navier–Stokes ligninger* i kompressibel form med inkludering af frie overflader. Ideen er at også modellere den del af oliefilmen, der eksisterer udenfor stempelringene. Ved tidsintegration af systemet kan indløbspunktets bevægelse følges.

Mens den første metode, der er baseret på Reynolds ligning, er begrænset af forudsætningen om stationære kørselsforhold, viser det sig at modellen baseret på Navier–Stokes ligninger er beregningsmæssigt kostbar. Forskellige løsningsmuligheder på dette problem, så som implicit tidsintegration og en kombination af Reynolds ligning og Navier–Stokes ligninger, undersøges.

Teksten indeholder detaljerede udledninger af de anvendte modeller, og de numeriske egenskaber fastlægges ved hjælp af konvergensstudier og sammenligninger med standardiserede testproblemer.

Acknowledgments

First I would like thank MAN Diesel A/S for offering this project. I would like to thank my main advisor on the company side Anders Vølund for following my work closely, and good guidance and discussions especially related to the Reynolds equation. I would also like to thank the Technical University of Denmark, Department of Mechanical Engineering (MEK) and my advisor there Peder Klit.

Besides from my main advisors many people have helped me along the way. I wish to thank Associate Professor Harry Bingham from MEK for his teaching in numerical fluid dynamics, and for introducing me to the method of Smoothed Particle Hydrodynamics. This opened my eyes to the world of meshfree methods, which I was not aware of before. Harry also provided me with a contact for my stay abroad, which I appreciated, although it was found that I should not go there.

Special thanks go to Jens Honoré Walther (Associate Professor at MEK, and also at ETH Zürich), who I meet in the middle of my studies. At that time my work was in a difficult phase, however, Jens provided me with help, guidance, and encouragement. The importance of his engagement for the successful development of my work, should not be underestimated. My stay abroad was initiated by Jens, who invited me to go to ETH in Zürich. There I met Mathias Dietzel who helped me with the description of the dynamics of triple points. I would also like to thank Michael Bergdorf, Ivo F. Sbalzarini, and the remaining people at the CoLab for useful discussions, practical help, and for making my stay at ETH very enjoyable.

Thanks go to Jan Hesthaven (Professor, Brown University) for help with implicit Runge–Kutta methods.

At MAN Diesel I would like to thank John M. Hansen (PhD) for useful tips and help with the `C++` programming language. Thanks go to Stefan Mayer (PhD) for discussions on numerical fluid mechanics. Also, I would like the thank the people from the development department for being nice colleagues.

At the Technical University of Denmark I would like the thank Klaus Kjølhede and Niels Heinrichson with whom I shared office. With Niels I had good discussions about numerical methods and the Reynolds equation. But as important, I wish to thank both of you for good times in your company. Also thanks to Allan Gersborg-Hansen for technical discussions as well as coffee-sessions.

Finally, but not least, I would like to thank my friends and family for being there during the hard times and the times of celebration.

Contents

1 Introduction **11**
1.1 The Industrial PhD-framework in Denmark 11
1.2 Presentation of MAN Diesel A/S 11
1.3 Presentation of the PhD project 13
1.4 How to read this document 27

I Reynolds equation **31**

2 The Reynolds equation **33**
2.1 Derivation of the Reynolds equation 33
2.2 Arbitrary solid body motion 39
2.3 Special case $h_b = 0$ 42
2.4 Special case $h_b = V_b = U_a = 0$ 43
2.5 Special case $h_b = V_b = V_a = 0$ 43
2.6 Special case $h_b = V_b = U_a = V_a = 0$, $\mu = \mu_0$ 45
2.7 Special case $h_b = V_b = U_a = V_a = 0$, $\mu = \mu_0$, $\rho = \rho_0$ 46
2.8 Obtaining pressure distribution 46
2.9 Obtaining flow rates 47
2.10 Obtaining velocity distributions 47

3 Boundary conditions for the Reynolds equation **49**
3.1 Cavitation 50

4 Discretization of the Reynolds equation **55**
4.1 Incompressible case 55
4.2 Compressible case 57
4.3 Sensitivity analysis 58
4.4 Cavitation iteration – general description 61
4.4.1 Equation of state 62
4.4.2 Internal boundary conditions 62
4.4.3 Greedy iterations 63
4.4.4 Newton iteration 64
4.4.5 Bisection method 65

4.5 Cavitation iteration – actual implementation 66

5 Results from the Reynolds equation 69
5.1 Fixed incline slider bearing 69
5.1.1 Analytical, incompressible, and compressible solution . . . 70
5.1.2 Velocity components 73
5.2 Conditions for backflow 74
5.3 Relating h_∞, h_{in}, and $h_{\min}$ 78

II Navier–Stokes equations 89

6 The Navier–Stokes equations 91
6.1 The Navier–Stokes equations 92
6.2 Equation of state 94
6.2.1 Ideal gas law 95
6.2.2 Monaghan power law 95
6.2.3 Dowson-Higginson law 96
6.2.4 Cavitation model 97

7 Multibody kinematics 101

8 Boundary conditions for the Navier–Stokes equations 105
8.1 Solid boundaries 105
8.2 Pressure boundary condition 106
8.2.1 Extrapolation 106
8.2.2 Neumann condition 107
8.2.3 Comparison of methods 108
8.3 Free surface 108
8.4 Triple point 111
8.4.1 Dynamic contact angle model 112
8.4.2 Zero shear stress model 114
8.4.3 Comparison of methods 114

9 Discretization of the Navier–Stokes equations 117
9.1 Discretization method 117
9.2 Moving Least Squares 118
9.2.1 Surface fitting 119
9.2.2 Taylor expansion 128
9.3 Neighbor search 130
9.4 Spacial derivatives 132
9.4.1 Sensitivity analysis 134
9.5 Governing equations 137
9.5.1 Gradients of the governing equations 138

9.6 Boundary conditions . . . 139
9.6.1 Residual equations . . . 140
9.6.2 Gradient of residual equations . . . 147
9.6.3 Enforcing boundary conditions . . . 153
9.6.4 Sensitivity analysis of explicit boundary conditions . . . 156
9.7 Time integration . . . 159
9.7.1 The state vector . . . 159
9.7.2 Explicit time integration . . . 161
9.7.3 Implicit time integration . . . 164
9.8 Remeshing . . . 169
9.8.1 Remeshing of solids . . . 172
9.8.2 Remeshing of free surfaces . . . 174
9.8.3 Remeshing of the fluid domain . . . 177
9.8.4 Interpolation of field variables . . . 180

10 Results from the Navier–Stokes equations 183
10.1 Lid driven cavity . . . 183
10.2 Fixed incline slider bearing . . . 189
10.3 Rotating free surface . . . 195
10.4 Drop oscillation . . . 199
10.5 Piston ring simulation I . . . 203
10.6 Piston ring simulation II . . . 207
10.7 Discussion and future work . . . 212

11 Conclusion 215

A Sensitivity of free surface normal and curvature 217

B Comparison of Jacobian matrices 225

C On the finite difference approximation of $\mathrm{d}h/\mathrm{d}x$ 241

Nomenclature

$\mathbf{A}$	generic matrix
A_{ij}	element at row i column j of matrix $\mathbf{A}$
$\mathbf{a}$	generic vector
a_i	element i of vector $\mathbf{a}$
$\mathbf{A}'$	derivative of matrix $\mathbf{A}$ with respect to some variable of interest
$\mathbf{a}'$	derivative of vector $\mathbf{a}$ with respect to some variable of interest
$\mathbf{e}_j$	the j-th canonical basis vector (a zero vector of appropriate size with a 1 at element j)
$(\cdot)^k$	iteration level k or time step k
$(\cdot)_i$	generic variable $(\cdot)$ evaluated at discrete point i
$\Delta(\cdot)$	finite increment of $(\cdot)$
$\dfrac{\partial f}{\partial(\cdot)}$	partial derivative of f with respect to $(\cdot)$
$\dfrac{\mathrm{d}f}{\mathrm{d}(\cdot)}$	total (*substantial*, *material*, or *Lagrangian*) derivative of f with respect to $(\cdot)$
$\mathbf{x}$	position vector
x	Cartesian x-coordinate
y	Cartesian y-coordinate
$\mathbf{u}$	velocity vector
u	x-component of velocity vector
v	y-component of velocity vector

U_a	Lagrangian u-velocity on upper part
V_a	Lagrangian v-velocity on upper part
u_a	Eulerian u-velocity on upper part
v_a	Eulerian v-velocity on upper part
U_b	Lagrangian u-velocity on lower part
V_b	Lagrangian v-velocity on lower part
u_b	Eulerian u-velocity on lower part
v_b	Eulerian v-velocity on lower part
x	x-position of solid body
y	y-position of solid body
ϕ	angular orientation of solid body
u	u-velocity of solid body
v	v-velocity of solid body
ω	angular velocity of solid body
a_x	x-acceleration of solid body
a_y	y-acceleration of solid body
a_ϕ	angular acceleration of solid body
$\mathbf{n}$	normal vector
n_x	x-component of normal vector
n_y	y-component of normal vector
$\mathbf{t}$	tangent vector
t_x	x-component of tangent vector
t_y	y-component of tangent vector
m	mass flow rate
q	volume flow rate
q_{cav}	volume flow rate at the onset of cavitation

p	pressure
p_{in}	inlet pressure
p_{out}	outlet pressure
p_{cav}	cavitation pressure
p_{amb}	ambient pressure from surroundings
x_{cav}	location of onset of cavitation
x_{in}	location of the inlet of a lubricated conjunction
$x_{\text{rebuildup}}$	location of rebuild-up of the oil film
h_{in}	oil film thickness at inlet of a lubricated conjunction
h_{∞}	oil film thickness on liner far away from piston ring
h_{min}	minimum oil film thickness in lubricated conjunction
a	speed of sound
ρ	density
μ	dynamic viscosity
κ	curvature
σ	surface tension
t	time
W	kernel function
h	smoothing length
O	order function or complexity
δ_{ij}	Kronecker delta
v_i	velocity tensor
σ_{ij}	stress tensor
τ_{ij}	shear stress tensor
ε_{ij}	deformation rate tensor
R_e	residual of density extrapolation

R_n residual of Neumann condition for density

U_f residual of no flow through solid condition

U_n residual of normal stress condition

U_s residual of shear stress condition

U_x residual of x-component of dynamic contact angle condition

U_y residual of y-component of dynamic contact angle condition

Chapter 1

Introduction

1.1 The Industrial PhD-framework in Denmark

This section is included for readers unfamiliar with the components of an Industrial PhD-project in Denmark.

In Denmark the PhD-title can be obtained after three years of research in a given field. Three years is the time frame for a regular PhD scholarship as well as Industrial PhD scholarships. The student must engage in different predefined activities, namely: research, teaching, publication, collaboration with foreign institutions, and a special course provided by the Ministry of Science Technology and Innovation. The company is free to choose the subject of the project, only it must be accepted by a board of judges from the Danish Academy of Technical Sciences. The company will pay two thirds of the salary for the student, while the Government pays the remaining third as well as fees for the hosting university. The student is supposed to spend 50% of his time at the company and the university, respctively.

1.2 Presentation of MAN Diesel A/S

The history begins 160 years ago in the year 1843, when a man named Baumgarten founded his shop in Copenhagen. From general purpose production the company soon started to focus on production of ships and steam engines. In 1854 another person named Burmeister joined in as partner. Later in 1865 the Englishman William Wain was brought to the company in order to contribute with the latest knowledge on steam engines. At the same time Baumgarten had his retirement and the company was renamed Burmeister & Wain (B&W).

At the end of the 19th century the Diesel engine was invented by the German Rudolf Diesel. B&W obtained a license for the Diesel engines as stand-alone manufacturer in Denmark. The company was evolving and in 1912 the B&W ship “Selandia” crossed the Atlantic Sea. This was the first time in history that

Figure 1.1: Container carrier fitted with MAN Diesel engine type 12K98ME-C.

a Diesel engine powered ship would accomplish that voyage.

In the 1950's the turbo charger was added to the engines, making it possible to produce even more horse power without increasing the engine size. This was necessary to stay competitive with steam turbines, which were also used for the propulsion of ships.

In the 1970's the World experienced an energy crisis. This meant the abandoning of steam turbine engines in ships (except for applications for gas tankers). The reason is that Diesel engines have a much better fuel economy than the steam turbine engines. Despite of the favorable change in market the 1970's were hard times for B&W due to international competition.

In the beginning of the 1980's B&W was split in two. The design office for Diesel engines was taken over by the German MAN Group, while the shipyard continued to be a unit on its own. However, competition from other shipyards increased in the following years, and finally the shipyard was forced to shut down some years later in the mid 1990's.

Today production activities in Denmark are small compared to what they used to be in old times. However, four stroke engines are still manufactured locally as well as special parts for the large two stroke engines. The office in Copenhagen is concerned with engine design and service operation for the clients. Thus from being a heavy industry the activities in Denmark are now mainly research and development, and operation. The company name was recently changed from MAN B&W Diesel A/S to MAN Diesel A/S (MD).

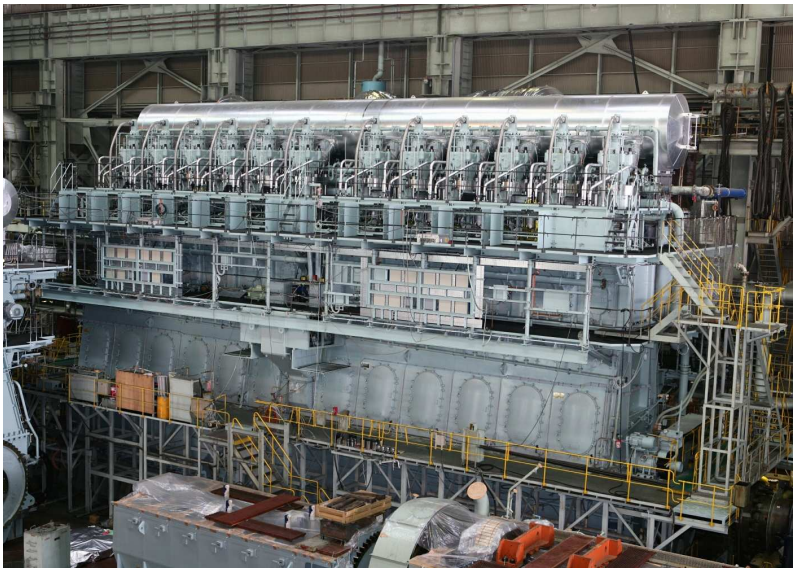

Figure 1.2: A MAN Diesel engine type 12K98ME-C. It has 12 cylinders each with a bore of 980 mm and a stroke of 2400 mm. The power output is 72240 kW.

The latest development features the electronically controlled engine, which has no cam shaft. Instead, fuel injection and exhaust valves are operated by electronically controlled fuel valves and hydraulic actuators. In figure 1.3 an x-ray picture of the 12K98ME-C engine is shown.

The business of MD is based on the sales of licenses for shipyards and engine manufacturers around the World, mainly in the Eastern countries Korea, Japan, and China. The company is holding about 80% of the market for some engine types, and must be considered the leading player in the field. The second business is service for running engines, such as engine overhaul and inspections. Finally, the third business component is the sales of original spare parts.

1.3 Presentation of the PhD project

Piston rings are vital components of any internal combustion engine. Their purpose is to act as sealing for the highly pressurized combustion gas, so that it cannot escape between the piston and the cylinder liner. They are manufactured from different kinds of steal alloys or cast iron. Typically the surface is coated with a special layer, which wears of little by little during running in. In figure 1.4 an x-ray view of a single cylinder is shown, and in figure 1.5 the piston ring assembly is shown with annotations. We describe the working principle of the

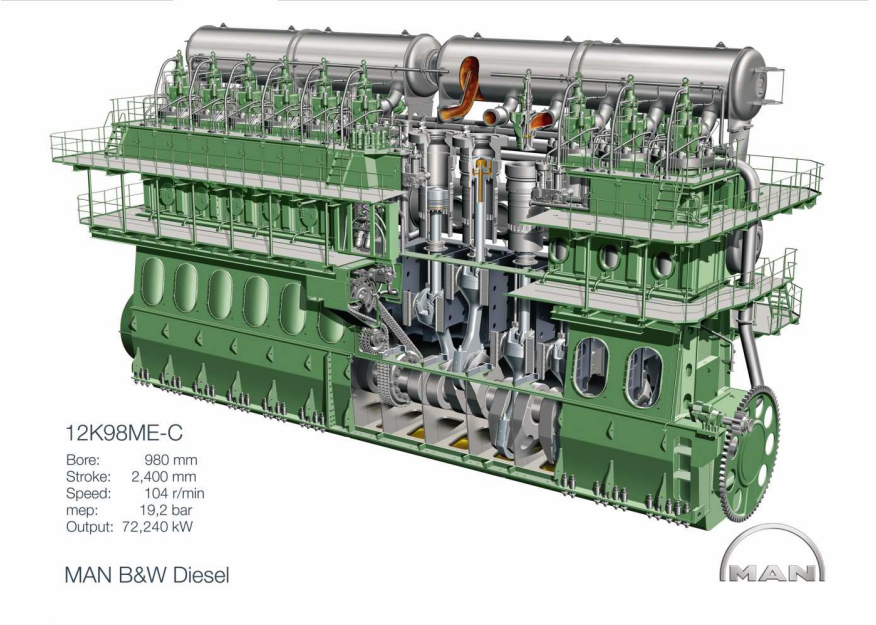

Figure 1.3: An x-ray view through an 12K98ME-C engine.

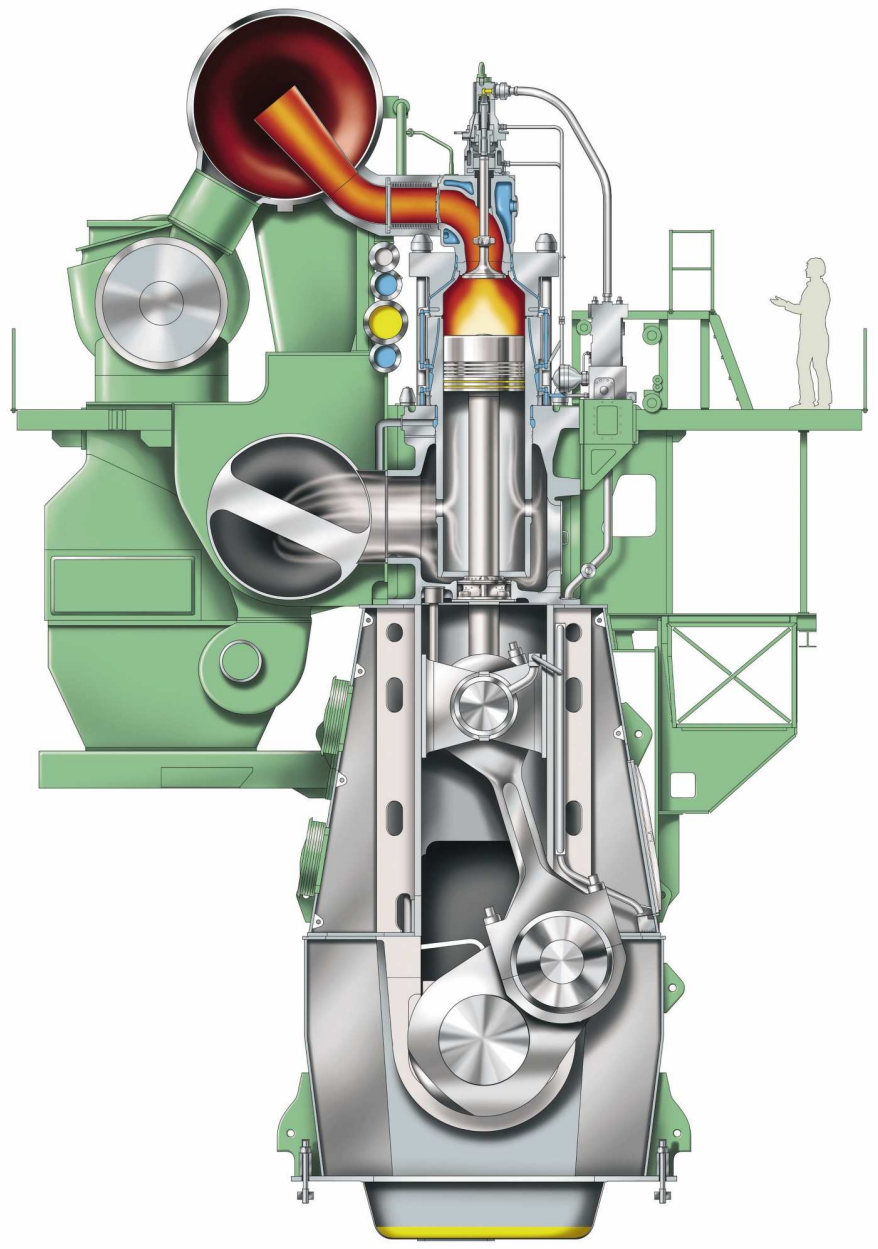

Figure 1.4: An x-ray view through a single cylinder of the 12K98ME-C engine.

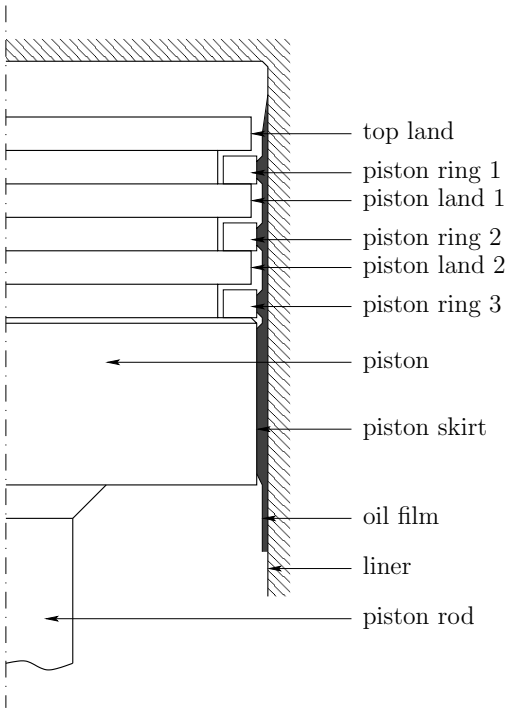

Figure 1.5: A view of the piston ring package.

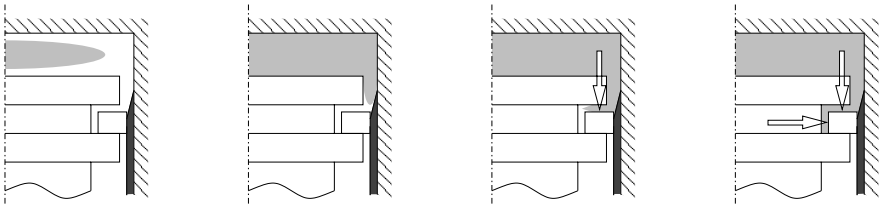

Figure 1.6: Working principle of a piston ring. Heavy shading represents the oil film. Light shading represents the expanding high pressure region from the combustion. The arrows represent forces on the piston ring from the gas pressure. Other forces (from the oil film, friction, etc) are not shown.

piston rings based on the combustion cycle, see figure 1.6. 1) The combustion takes place which highly increases the pressure in the combustion chamber, 2) The pressure wave travels through the gap between the liner and the piston top land, 3) High pressure reaches the piston ring, where it travels through the small gap (about 1/10 of a millimeter) between the piston ring and the piston, 4) Finally it reaches the back side of the piston ring where it results in a force, that pushes the piston ring onto the liner, 5) Ideally the oil film will generate a counter force, which prevents the piston ring from touching the liner. A similar behavior takes place during the compression cycle.

In large engines more than one piston ring is fitted – typically four. Each piston ring has a number of grooves where the gas can escape. This is done in order to distribute the total pressure drop to more than one ring. Figure 1.7 shows a picture of the piston ring package. The picture is taken during inspection of the engine.

The lubrication of piston rings is a very delicate matter. On one hand we want to reduce the lubrication oil consumption, on the other hand we want enough oil so that wear of the piston rings and the liner can be as small as possible. Here it should be noted that large two-stroke engines differ from the familiar small-size four-stroke engines used in e.g. automobiles. In the latter case oil is constantly splashed from the oil sump onto the cylinder walls. A special oil control ring makes sure that only a predefined amount of oil remains on the liner after the downward movement of the piston. In this way oil consumption can be controlled accurately. However, two-stroke engines have no oil sump and must therefore be lubricated in a different way. For large two-stroke engines this is done by injecting oil through holes in the cylinder wall. Clearly, an excess of oil will simply be burned off. Therefore one tries to find the correct rate of lubrication, balancing consumption and sufficient lubrication. Since no oil sump exists no oil control ring is fitted.

Investigations show that piston rings experience different kinds of lubrication mechanisms 1) On the middle part of the stroke, where piston speed is high,

Figure 1.7: A close up view of the piston ring package through the Scavenging ports at the bottom dead center. Note the pressure relieve groove on the top ring.

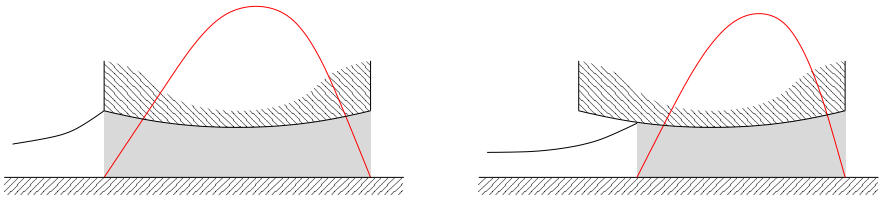

Figure 1.8: Left: A fully flooded piston ring. Right: A piston ring under starved running conditions. The shaded region indicates the area where Reynolds equation applies. The pressure curves (red) are merely graphics and are not related.

the hydrodynamic action results in an oil film fully separating the piston ring from the liner, 2) Approaching either end of the stroke the piston speed becomes smaller and the oil film now relies on the squeeze film action. At some point the film becomes so thin that the surface roughnesses (asperities) of the piston ring and the liner touch each other. This regime is called mixed lubrication. 3) Finally, near the dead centers the piston speed is very small (at the dead centers it is zero, since the direction of motion is reversed). Here the piston rings will experience boundary lubrication, that is, direct contact between the parts exist.

The research on piston ring behavior has a long history, virtually beginning at the birth of the internal combustion engine (The earlier steam engines, also with pistons, had a leather sealing instead of piston rings. Leather is, however, not suited as a replacement for piston rings.) Many experimental as well as theoretical investigations have been carried out. This project takes the theoretical approach in order to achieve new results and knowledge. The basis for the theoretical treatment of thin fluid films is the *Reynolds equation*. Without going into details we will discuss some of the features of this equation (the first chapters are devoted to a detailed treatment). Consider the two situations shown in figure 1.8. The figure shows a section along the liner, so one should imagine the piston ring to expand out of the paper and also into the half space on the backside of the paper.

The left part of the figure shows a situation known as *fully flooded* running conditions. In this situation the running part of the piston ring is fully wetted with oil, giving the opportunity to build up oil pressure forces on the whole part of the piston ring. A pressure curve is sketched for the situation where squeeze action is predominant (the piston ring is moving closer to the liner). This pressure distribution can be found by solving Reynolds equation for that system. Note that the domain where Reynolds equation applies (the shaded area) is well defined being equal to the total width of the piston ring.

The right part of figure 1.8 shows a situation denoted as *starved* running conditions. It is seen that only part of the running surface of the piston ring is wetted with oil. Again we have sketched a possible pressure curve for the situation where the piston ring approaches the liner. Now, this pressure curve

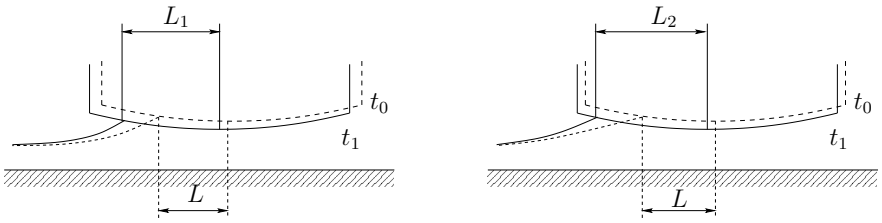

Figure 1.9: Two examples of oil configuration outside the piston ring. The broken lines denote piston ring position and oil film at time $t = t_0$. The next time instant $t = t_1$ is displayed with full lines. Note that $L_1 \neq L_2$ even though L is shared for the left and the right configurations. The difference in L_1 and L_2 is due to the difference of oil film thickness outside the piston ring.

can also be found by applying Reynolds equation to the shaded area. However, an important difficulty arises for the computation of how the system evolves. How does the location of the contact point between air, oil, and piston ring (the so-called triple point) move due to movement of the piston ring? In figure 1.9 we have shown two scenarios of the oil configuration on the liner, and sketched how the triple point changes position on the piston ring. It is seen that depending on the amount of oil *outside* the area where Reynolds equation is applicable we get different results. This effect is the starting point for this project. Stated more precisely the problem definition is given as

> Establish the boundary conditions for Reynolds equation, such that the domain where Reynolds equation is to be applied is known at all times. Specifically the case of starved running conditions must be handled.

Current simulation programs for piston ring analysis circumvent the problem described above by making two assumptions 1) the oil film thickness on the liner is constant (set to some nominal value), 2) the build up of the oil film thickness in front of the piston ring is constant. We will discuss these two assumptions in the following. In figure 1.10 a sketch of oil build up in front of the piston ring is shown.

It is seen that the film thickness experienced by the piston ring is bigger, than the undisturbed film thickness on the liner. This effect comes from the no-slip condition on the liner and on the piston ring in combination with continuity of flow. In figure 1.10 we have taken the piston ring to be stationary and the liner to be moving. This is just a matter of choosing coordinate systems. It is seen that a block of oil traveling with the liner is slowed down on the top edge by the piston ring. Clearly, to maintain continuity of flow (in a steady state situation) the parabolic-like shaded area must equal the shaded area of the block. Therefore *build up* of oil takes place. A usual assumption is that h_{in} is related to h_∞ by the

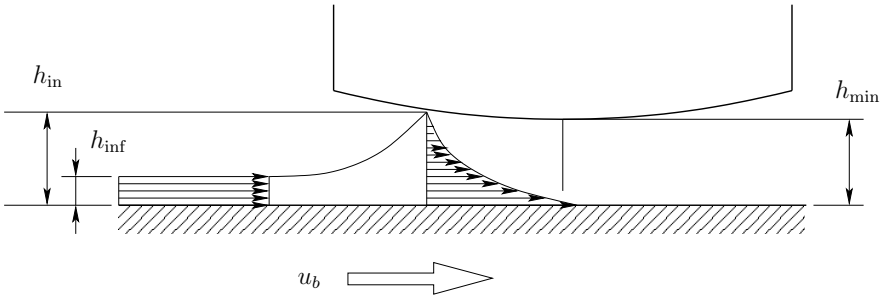

Figure 1.10: Build up of oil in front of piston ring.

following ratio

$$\frac{h_{\text{in}}}{h_\infty} = 2$$

This could be true for some situations (e.g. for an appropriate pressure drop over the piston ring), but in general this ratio is not constant. However, it is a usual procedure to keep the ratio constant and combine it with a constant nominal film thickness h_∞. When these quantities are fixed one can determine h_{in}, which then also becomes a constant. It should be noted that such a setup would still allow the simulation of starvation, since the minimum film thickness under the piston ring h_{min} is not constant. It depends on the loading of the piston ring due to e.g. gas forces on the backside of the ring. However, as already mentioned the inlet film thickness h_{in} becomes a constant with this setup. The aim of this work is to overcome that restriction.

Other workers in the field have addressed the problem of determining the inlet location. In the paper by Esfahanian [1] a single ring is treated assuming fully flooded conditions at all times. Dowson [2] analyzes a single ring and a complete ring pack. In that paper the undisturbed oil film thickness is assumed to be one half of the film thickness at the location under the piston ring where $\partial p/\partial x = 0$. A more complicated model is given in Han [3], where oil build up in front of the piston ring is modeled by a trapezoidal geometry. The same idea is followed in Gamble [4] except the build up is assumed to have a parabolic shape.

Choice of strategy

In order to overcome the assumptions mentioned in the previous section, it was decided to expand the simulation domain beyond the domain of the Reynolds equation. The idea is to keep track of the oil film also outside of the actually lubricated conjunction. This will require the simulation of the free surface outside the piston ring.

The Reynolds equation is not applicable for free surface flows. Therefore another set of equations must be used, at least for the part of the domain where a free surface exist. We have decided to used the Navier–Stokes equations for the free surface part. Since the Reynolds equation is "contained" (derived from) the Navier–Stokes equations it is tempting to use the Navier–Stokes equations everywhere in the domain – also under the piston ring. In fact we have tried both options, that is, 1) using the Navier–Stokes equations everywhere, and 2) combining the Navier–Stokes and the Reynolds equation on different parts of the simulation domain.

A model for the dynamics of the triple point movement is also needed. If we know the position of the triple point at all times, then it is a simple matter to calculate the inlet film thickness as well as the starting point for the Reynolds equation.

In the following sections we discuss the methodologies used to solve the Reynolds equation as well as the Navier–Stokes equations.

Choice of method for the Reynolds equation

As will been seen later (section 2.1) the Reynolds equation for a 2D physical flow problem is only 1D in the mathematical sense. This makes it particularly easy to solve by e.g. the finite difference method. Under certain assumptions (incompressibility and constant viscosity) we can even obtain analytical solutions. Reynolds equation is traditionally discretized using the finite difference method, but other methods such as the finite element method can also be employed. In this thesis we have chosen to use the finite difference method, because it is efficient and quite easy to implement.

Choice of method for the Navier-Stokes equations

The simulation of free surface flows is a demanding task, because the free surface boundary evolves as the simulation goes on. A number of different formulations for free surface flows exist. They may be grouped into two major categories 1) two phase formulations, and 2) single phase formulations. The difference between these methods is shown in figure 1.11. The two phase model is characterized by having a constant and predefined simulation domain. Some parts of the domain contain the liquid phase, while the other part contains the gas or void phase. An example of this formulation is the VOF method [5]. The single phase formulation on the other hand includes only the liquid domain in the simulation domain. Therefore the simulation domain changes as the computation goes on. Because the single phase method does not simulate the gas phase, it must be considered as a void. Thus a single phase method cannot model e.g. frictional forces on the free surface boundary. If this is needed then both phases must be modeled.

For this application we have chosen to use the single phase method. The main

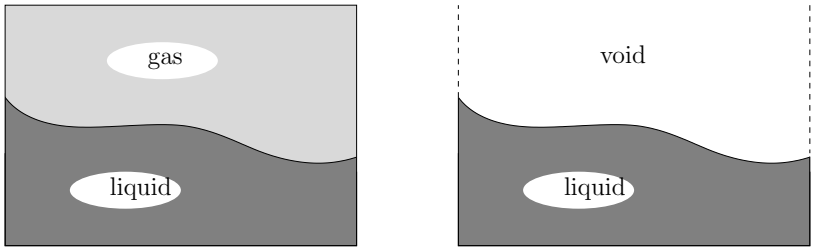

Figure 1.11: Simulation of a free surface. Left: Two phase system. Right: Single phase system.

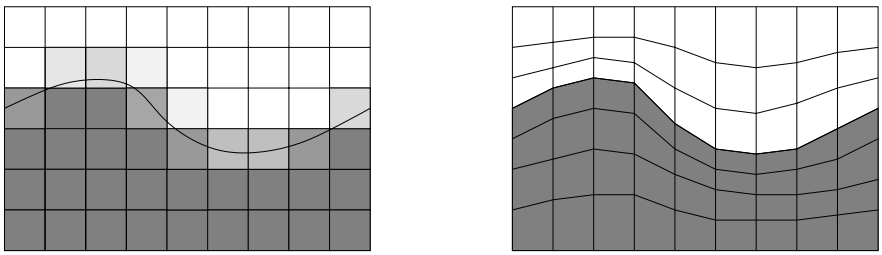

Figure 1.12: Left: Interface capturing method. Right: Interface tracking method.

reason is that we consider the friction from the gas phase to be negligible. Another reason, which is also important, is that choosing the single phase formulation will reduce the overall problem size because only the liquid domain is included in the model.

The class of modeling methods for handling the free surface itself may also be divided into two categories in the following way 1) surface capturing methods, and 2) surface tracking methods. The methods are indicated in figure 1.12. A surface capturing method requires that we use a domain that is bigger than the liquid domain. The idea is to create a discretization that can handle discontinuities. The free surface will then be represented by a jump in e.g. density, and can for example be evolved using a level-set formulation. A surface tracking method on the other hand deforms the mesh or nodal layout so that it follows the free surface at all times. For large deformations remeshing becomes necessary with this method. It is noted that the surface tracking method can be used with the two phase formulation as well as the single phase formulation.

In this work we have decided to use the surface tracking method. In fact this is our only possible choice, since we have already decided on the single phase formulation.

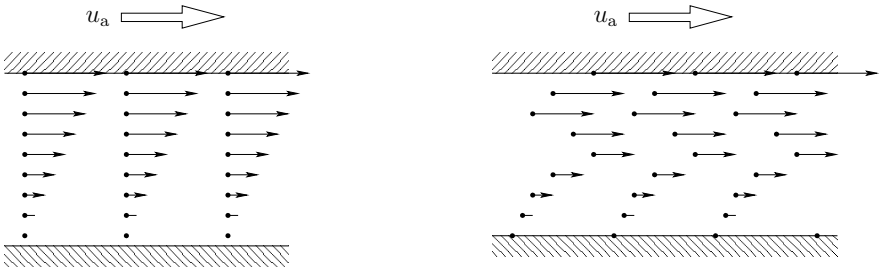

Figure 1.13: Couette flow between two infinite plates. Top plate moving with constant speed, lower plate stationary. Left: Eulerian coordinates (the computational nodes remain fixed). Right: Lagrangian coordinates (the computational nodes follow the flow field).

Having settled the methodology for the free surface representation we now turn our attention to the choice of what type of coordinates we will use. There are basically two types available 1) Eulerian coordinates, and 2) Lagrangian coordinates. In figure 1.13 we have indicated the difference with a simple example. The usual choice for fluid dynamics is the Eulerian coordinates. With this formulation the discretization points remain fixed in space. This is very convenient for many applications, such as the simulation of flow past an air-foil. In contrast the Lagrangian coordinates follow the deformation of the medium with which they are associated. This makes them the natural choice for solid mechanics. It should be mentioned that an intermediate method called the Arbitrary Euler Lagrange (ALE) method also exist. In this method the movement of the discretization points does not necessarily follow the deformation of the media, but can follow any user-defined deformation.

For this work we have chosen the Lagrangian coordinates. The reason for doing this is that the shape of the free surface will then be spanned automatically. There will be no *kinematic condition* on the flow at the free surface, as when using the Eulerian coordinates. Also, the piston rings are not stationary so fluid-solid interaction will take place. Moving boundaries are more easily handled with Lagrangian coordinates (An exception is when the velocity is parallel to the boundary. This case is equally well handled by the Eulerian approach.)

Now we are ready to decide on the over all discretization method. We are going to solve the 2D Navier–Stokes equations. The number of discretization methods available is big. An incomplete list includes

1. finite difference method (FDM) [6]
2. finite volume method (FVM) [6]
3. finite element method (FEM) [7]

4. conservation element - solution element method (CE/SE) [9]

5. smoothed particle hydrodynamics (SPH) [10]

Some of these methods are by nature more or less prepared for either Eulerian or Lagrangian coordinates. Most of them are based on a *mesh*, meaning that the computational nodes have a predefined connectivity. One method in the list above is special because it requires no mesh and is typically used with Lagrangian coordinates – the SPH method. SPH was invented in the 1970's and is therefore still relatively new. It was originally designed for the simulation of stars and other phenomena in outer space. However, SPH and its descendants have also been used for fluid mechanics as well as for solid mechanics.

Despite the attractiveness due to simplicity and computational efficiency we have chosen not to use the SPH method. This is mainly because of theoretical results, that show that the method is not even 0 order consistent [11] [12]. Furthermore, experiments with a sample code showed rather poor results for the lid driven cavity problem[1]. The steady state solution was found, but the evolution from the initial configuration was not physically appealing. This was probably due to certain "fixes" needed to stabilize the method[2] [13].

Instead we have chosen a method in the family of the Moving Least Square methods [15]. These methods can be viewed as a corrected SPH (also closely related are the Reproducing Kernel methods) or as a generalization of the finite difference method. These methods boil down to a procedure for locally fitting a surface (spanned by appropriately chosen basis functions) to given data points. Based on these local approximations spacial derivatives can be obtained, which are then substituted into the governing equations. If the system evolves in time a time integration method such as Leapfrog or a Runge Kutta method can be placed on top. This is known as the semi-discretization technique.

Is it noted that the Moving Least Squares method, as well as SPH and other related methods, are applied to the (partial) differential equations in strong form. This makes the transformation of the continuous differential equations into a discrete system relatively easy. Also, since no integration step is needed as in weak form methods the possibility of *meshfree* formulations exist. To be meshfree means that no connectivity between the nodes needs to be defined. This means that "remeshing" becomes much more tractable (with respect to cost of generating a mesh without bad elements) because no elements are defined. However, nodes that are close to each other must still be detected. One could think of a

[1]The lid driven cavity problem is a standard benchmark for fluid flow solvers. See section 10.1 for details.

[2]In reference [13] a sample SPH code is provided together with an input file for the lid driven cavity problem. Trying it out, reveals that the simulation is not stable without using the so-called "Monaghan velocity averaging". However, this averaging has a severe impact on the transient response from initially at rest to fully developed flow. A simple visual inspection is disappointing.

meshfree method as a method that allows overlapping elements. This interpretation actually gives a pointer to the main drawback of these methods – namely that conservation is not obtained automatically, as it for example is in the finite volume method. It should be noted that SPH is special in this regard, since it *is* conservative although it is a strong form method. As mentioned earlier the method is unfortunately not consistent.

Summarizing we have chosen to employ the semi-discretization technique for the Navier–Stokes equations. The spacial derivatives are obtained using the Moving Least Squares method, and the time integration is performed by a Runge–Kutta method.

Evolution of the project

In the previous sections we have presented the considerations on which this work is founded. Also we have given details about the choice of discretization methods for the equations we are solving. However, in order to appreciate the remainder of this document a few words on the development of the project is in place.

In the beginning of this work focus was on the Reynolds equation, which is the natural starting point for lubrication theory. Small problems involving a single piston ring was examined in order to get some feeling for piston ring behavior. At a relatively early point in time we tried to resolve the inlet problem (the determination of h_{in} based on Reynolds equation only). However, these attempts were not very successful, especially from an academic point of view. There were too many assumptions and "sloppiness".

It was therefore decided to switch to the 2D Navier–Stokes equations and model also the free surface outside the piston rings. The first test problems to be solved were the Couette flow problem and the Poiseuille flow problem. After this another standard problem, the lid driven cavity problem, was used as a test case. Then came the implementation of the free surface boundary condition and a condition for the movement of the triple points. After this, only one thing was missing in order to simulate piston ring behavior – the effect of cavitation. Cavitation is a usual phenomenon that occurs in liquids when the pressure becomes smaller than the evaporation pressure of the liquid at the current temperature. Cavitation can be modeled in different ways, one of them being by a modification of the equation of state and the viscosity (section 6.2.4).

Unfortunately trouble came along when we tried to solve real life problems. Typically the oil film thickness is around 3–20 μm (Actually no oil film might be available at all, but we could choose 3 μm as a lower limit for the continuum model of the oil film.) Using a discretization of at least 5 five nodes in the direction across the oil film gives a spacial resolution of 0.6–4 μm. This puts a heavy restriction on the allowed step size for the time integration. Furthermore the stroke of the piston in large Diesel engines can be up to approximately 2.5 m. Clearly, the number of nodes and the number of time steps needed for a single

revolution of the crankshaft becomes excessive. Realizing this fact was a great disappointment, especially because it could actually have been foreseen.

As an attempt to overcome the problem of very small time steps an implicit Rung–Kutta method was implemented as a replacement for the explicit methods used so far. It is somewhat more complicated to use implicit methods because a system of equations (or multiple systems) must be solved at very time step. However, the maximum step size was increased by the implicit time integration method. Unfortunately the increase in step size was not big enough to counterweight the extra cost of solving the systems of equations associated with the implicit method. Actually, the computing time needed to reach some simulation time t_1 was increased with the implicit method compared to the explicit time integration method.

As a final way of saving computational effort the Navier–Stokes equations were combined with the Reynolds equation. The idea was to reduce the number of nodes by discretizing the domain under the piston ring mainly using the Reynolds equation. Only the part under the piston ring close to the free surface would then be modeled by the Navier–Stokes equations. This mixing of formulations turned out to be possible, however, it was not enough to cut down the simulation time by the amount needed.

At this rather late point in the project we were running out of good ideas. After agreement with my advisors it was decided to change the expectations for the outcome of the project. Instead of a 2D code with allot of physical modeling and time evolution, it was decided to focus on steady state solutions. This class of problems is considerably more simple than the time dependent one. In fact it turned out that the information needed to determine the inlet location could be obtained using the Reynolds equation only (section 5.3).

This evolution of the project is not directly reflected in the organization of this document. Instead the various sections have been grouped by topic. Some deviation from this overall strategy has been is made so that the document is as suited as possible for consecutive reading from page one. The next section provides the organization of the text.

1.4 How to read this document

This monograph is divided into two main parts. The first part deals with the Reynolds equation while the second part deals with the Navier–Stokes equations. Each part has the chapters 1) Governing equations, 2) Boundary conditions, 3) Discretization, and 4) Results. Below is a short summary of each chapter.

Chapter 1 is this introduction.

Chapter 2 contains a derivation of the Reynolds equation for 2D problems. Special attention is given to the relationship between Eulerian and Lagrangian velocity components. Specializations of the Reynolds equation for various stan-

dard cases are derived. The chapter ends with expressions for post processing in order to obtain flow rates and velocity fields.

Chapter 3 discusses the boundary conditions for the Reynolds equation. The main theme is cavitation, which can be handled in different ways. However, the equation always becomes nonlinear when cavitation takes place.

Chapter 4 is devoted to the discretization of the Reynolds equation using finite differences. Emphasis is on iterative methods for handling cavitation. We also provide a sensitivity analysis of Reynolds equation with respect to perturbation of the inlet (or outlet) point.

Chapter 5 contains results from solutions of the Reynolds equation. The fixed incline slider bearing is used as a test case to establish the convergence properties of the finite difference scheme. Illustrations of the velocity distribution within the bearing are given. After this we carry out an analytical investigation of the conditions under which backflow can occur in a lubricated conjunction. Finally, we examine further the relation between the oil film thickness on the liner, at the inlet, and the minimum thickness under the piston ring for steady state conditions. A concrete example is given using piston kinematics and pressure drops from a specific MD engine.

Chapter 6 marks the beginning of the second part. The Navier–Stokes equations are stated in the form relevant for this work. Also, different equations of state are given.

Chapter 7 contains expressions from multibody kinematics. These expressions are used to relate the velocity vectors that exist on the boundary of a rigid solid body to the motion of the body.

Chapter 8 discusses the boundary conditions for the Navier–Stokes equations. First the boundary conditions at solid boundaries are stated. Then two different formulations for the density (or pressure) on the boundary of the fluid domain are given. The free surface boundary condition is developed. Finally, two different methods for the triple point dynamics are given.

Chapter 9 deals with the discretization of the Navier–Stokes equations. This chapter is quite long. First we introduce the Moving Least Squares method by a data fitting procedure. An alternative derivation based on Taylor series is also given. We then discuss an important step of meshfree methods – searching for neighboring nodes. After this we carry out a sensitivity analysis of the Moving Least Squares method. These expressions are needed if an implicit time integration method is to be used.

Then follows the governing equations and boundary conditions in discrete form. Again we provide the derivatives of these terms, which are needed by an implicit time integration method. Two different ways of imposing the boundary conditions are considered.

Once the spacial discretization has been presented, we continue with the time integration methods. As already mentioned we have implemented an explicit as well as an implicit Runge–Kutta scheme. Details on how to obtain the Jacobian

matrix of the system are provided.

Finally, we discuss remeshing (redistribution) of the nodal positions. Although the Lagrangian formulation has some adaptive properties where will be areas in the solution domain where nodes spread or clump together. Therefore it is necessary to redistribute the nodes from time to time.

Chapter 10 contains simulation results from a number of test problems. These problems are used to investigate the convergence properties and time step limitations of the Navier–Stokes solver. Simulations of the oil film in piston ring lubrication are given, and finally suggestions for the future work are included.

Chapter 11 holds the conclusions from this work.

Part I

Reynolds equation

Chapter 2

The Reynolds equation

In this chapter we present the Reynolds equation for thin fluid films. First we provide a derivation of the equation, and then specializations of particular interest are presented. This chapter also includes two sections with various expressions relevant for post processing.

In this thesis we have only considered 2D flow problems. Therefore we derive the Reynolds equation for the 2D situation only. The full 3D Reynolds equation can be obtained by simply copying and adding terms in x, and replace x with the symbol for the missing coordinate direction.

2.1 Derivation of the Reynolds equation

The Reynolds equation is derived from the Navier–Stokes equations by non-dimensionalizing the system and then performing an analysis of the magnitudes of the coefficients. Keeping only the most significant terms corresponds to the following assumptions

- Pressure is constant across the film
- Inertia terms are neglected
- Bulk viscosity is neglected

Furthermore the following assumptions are made

- Density is constant across the film
- Viscosity is constant across the film

We do not include the order of magnitude analysis in our derivation but refer the interested reader to [23]. Instead we have indicated the impact of the assumptions in figure 2.2. This figure shows a fixed control volume and the stress components, that are not discarded from the Navier–Stokes equations, due to the order of

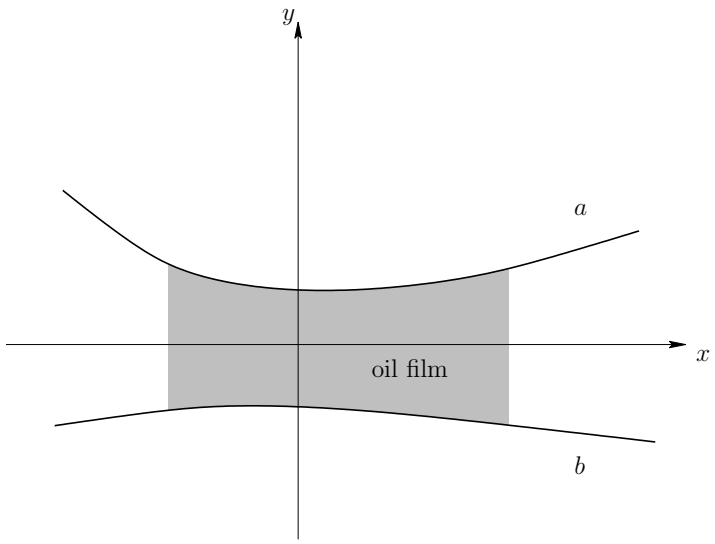

Figure 2.1: Coordinate system for Reynolds equation. The upper part of the bearing is denoted a, and the lower part is denoted by b.

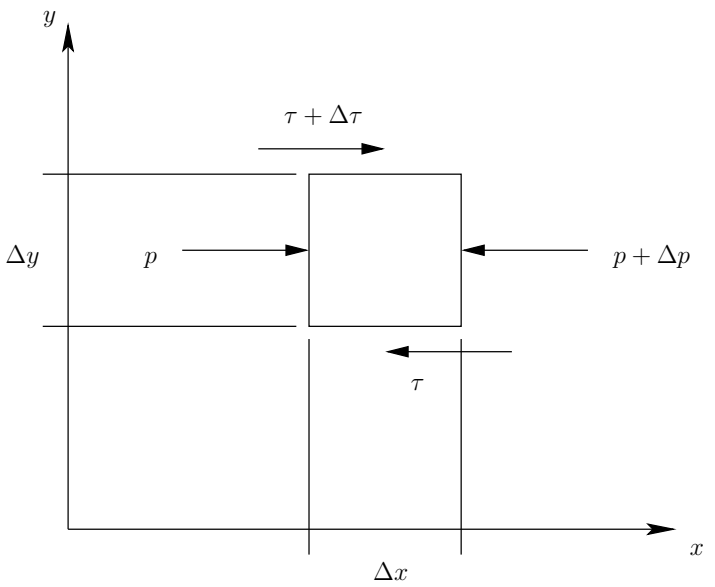

Figure 2.2: A control volume with acting stresses.

magnitude analysis. Coordinate direction x is along the fluid film, and coordinate direction y is oriented across the fluid film, figure 2.1. From figure 2.2 the force equilibrium becomes

$$\begin{aligned} p\Delta y + (\tau + \Delta\tau)\Delta x &= (p + \Delta p)\Delta y + \tau\Delta x \\ \Delta\tau\Delta x &= \Delta p\Delta y \end{aligned} \tag{2.1}$$

Making all changes Δ infinitely small gives

$$\frac{\partial p}{\partial x} = \frac{\partial \tau}{\partial y} \tag{2.2}$$

For a Newtonian fluid the relation between shear stress and shear rate is given by

$$\tau = \mu \frac{\partial u}{\partial y} \tag{2.3}$$

Inserting this in (2.2) gives

$$\frac{\partial p}{\partial x} = \frac{\partial}{\partial y}\left(\mu \frac{\partial u}{\partial y}\right) \tag{2.4}$$

In this equation $u = u(x, y, t)$ is the velocity component of the fluid in the x-direction, evaluated in the point (x, y) at time t. The notation $u = u(x, y, t)$ reflects that u is a Eulerian velocity component.

It is assumed that the pressure is constant across the film thickness, which means

$$\frac{\partial p}{\partial y} = 0 \tag{2.5}$$

In order words the pressure does not depend on y. In the same manner it is assumed that density and viscosity do not depend on y. Written with symbols

$$\begin{aligned} p &\neq p(y) \\ \rho &\neq \rho(y) \\ \mu &\neq \mu(y) \end{aligned} \tag{2.6}$$

Thus (2.4) may be integrated directly with respect to y. One has

$$\begin{aligned} \int \frac{\partial p}{\partial x} \mathrm{d}y = \int \frac{\partial}{\partial y}\left(\mu \frac{\partial u}{\partial y}\right) \mathrm{d}y &= \int \mathrm{d}\left(\mu \frac{\partial u}{\partial y}\right) \\ \frac{\partial p}{\partial x} y + C_1 &= \mu \frac{\partial u}{\partial y} \end{aligned} \tag{2.7}$$

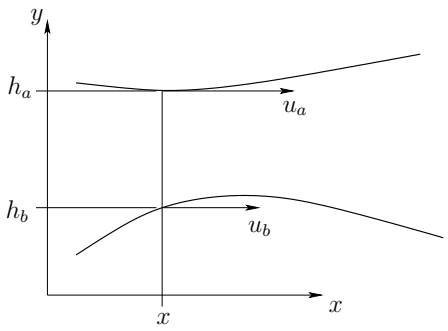

Figure 2.3: Two arbitrary surfaces separated by an oil film.

Another integration gives

$$\frac{1}{2}\frac{\partial p}{\partial x}y^2 + C_1 y + C_2 = \mu u \tag{2.8}$$

Consider a cross-section of an oil film separating two bodies denoted a and b, figure 2.3. Quantities $h_a = h_a(x,t)$ and $h_b = h_b(x,t)$ denote the distance from the x-axis at point x at time t. Quantities $u_a = u_a(x,t)$ and $u_b = u_b(x,t)$ denote the (Eulerian) velocity components on the bodies along the x-axis at point x at time t. The boundary conditions for the oil film are then

$$\begin{aligned} u = u(x,y,t): \quad & u(x,h_a,t) = u_a(x,t) \\ & u(x,h_b,t) = u_b(x,t) \end{aligned} \tag{2.9}$$

Constants of integration C_1 and C_2 are determined by inserting the boundary conditions in (2.8), giving two equations

$$\begin{aligned} \frac{1}{2}\frac{\partial p}{\partial x}h_a^2 + C_1 h_a + C_2 &= \mu u_a \\ \frac{1}{2}\frac{\partial p}{\partial x}h_b^2 + C_1 h_b + C_2 &= \mu u_b \end{aligned} \tag{2.10}$$

The solution is

$$\begin{aligned} \frac{1}{2}\frac{\partial p}{\partial x}(h_a^2 - h_b^2) + C_1(h_a - h_b) &= \mu(u_a - u_b) \\ \frac{1}{2}\frac{\partial p}{\partial x}(h_a^2 h_b - h_b^2 h_a) + C_2(h_b - h_a) &= \mu(u_a h_b - u_b h_a) \end{aligned} \tag{2.11}$$

or

$$\begin{aligned} C_1 &= \frac{u_a - u_b}{h_a - h_b}\mu - \frac{1}{2}\frac{\partial p}{\partial x}(h_a + h_b) \\ C_2 &= \frac{u_a h_b - u_b h_a}{h_b - h_a}\mu + \frac{1}{2}\frac{\partial p}{\partial x}h_a h_b \end{aligned} \tag{2.12}$$

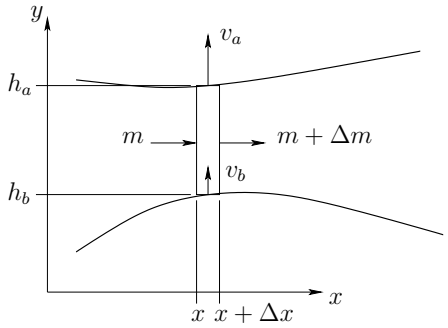

Figure 2.4: A control volume for mass flow.

The u-velocity can be obtained from manipulation of (2.8) and substitution of (2.12)

$$\begin{aligned} u = u(x,y,t) &= \frac{1}{2\mu}\frac{\partial p}{\partial x}y^2 + \frac{1}{\mu}C_1 y + \frac{1}{\mu}C_2 \\ &= \frac{1}{2\mu}\frac{\partial p}{\partial x}\left(y^2 - (h_a + h_b)y + h_a h_b\right) \\ &+ \frac{u_a - u_b}{h_a - h_b}y + \frac{u_a h_b - u_b h_a}{h_b - h_a} \end{aligned} \tag{2.13}$$

The mass flow rate m through a section can be obtained similarly by

$$\begin{aligned} m = m(x,t) &= \int_{h_b}^{h_a} \rho u \mathrm{d}y \\ &= \int_{h_b}^{h_a} \rho\left(\frac{1}{2\mu}\frac{\partial p}{\partial x}y^2 + \frac{1}{\mu}C_1 y + \frac{1}{\mu}C_2\right)\mathrm{d}y \\ &= \frac{\rho(h_a^3 - h_b^3)}{6\mu}\frac{\partial p}{\partial x} + \frac{\rho(h_a^2 - h_b^2)}{2\mu}C_1 + \frac{\rho(h_a - h_b)}{\mu}C_2 \end{aligned} \tag{2.14}$$

Substitution of (2.12) gives, after some manipulation

$$\begin{aligned} m = &\left[\frac{\rho(h_a^3 - h_b^3)}{6\mu} - \frac{\rho(h_a^2 - h_b^2)(h_a + h_b)}{4\mu} + \frac{\rho(h_a - h_b)h_a h_b}{2\mu}\right]\frac{\partial p}{\partial x} \\ &+ \frac{\rho(h_a - h_b)(u_a + u_b)}{2} \end{aligned} \tag{2.15}$$

In figure 2.4 we have shown a control volume with mass flow indicated at the four edges. Continuity of mass gives

$$m - (m + \Delta m) + \rho v_b \Delta x - \rho v_a \Delta x = (h_a - h_b)\Delta x \frac{\Delta \rho}{\Delta t} \tag{2.16}$$

where

$$\begin{aligned} v_a &= v_a(x,t) = \frac{\partial h_a}{\partial t} \\ v_b &= v_b(x,t) = \frac{\partial h_b}{\partial t} \end{aligned} \tag{2.17}$$

are the (Eulerian) velocity components in the y-direction of the horizontal edges of the control volume, evaluated at point x at time t. In the limit of small Δ one has

$$-\frac{\partial m}{\partial x} + \rho(v_b - v_a) = (h_a - h_b)\frac{\partial \rho}{\partial t} \tag{2.18}$$

Substitution of the expression for m (2.15) gives *Reynolds equation*

$$\begin{aligned} &\frac{\partial}{\partial x}\left(\left[\frac{\rho(h_a^3 - h_b^3)}{6\mu} - \frac{\rho(h_a^2 - h_b^2)(h_a + h_b)}{4\mu} + \frac{\rho(h_a - h_b)h_a h_b}{2\mu}\right]\frac{\partial p}{\partial x}\right) \\ &\quad = -\frac{\partial}{\partial x}\left(\frac{\rho(h_a - h_b)(u_a + u_b)}{2}\right) + \rho(v_b - v_a) - (h_a - h_b)\frac{\partial \rho}{\partial t} \end{aligned} \tag{2.19}$$

This is the Reynolds equation in a general setting for 2D problems. The dependencies are repeated

$$\begin{aligned} x &\text{ independent variable} \\ t &\text{ independent variable} \\ \rho &= \rho(x,t) \\ \mu &= \mu(x,t) \\ h_a &= h_a(x,t) \\ h_b &= h_b(x,t) \\ u_a &= u_a(x,t) \\ u_b &= u_b(x,t) \\ v_a &= v_a(x,t) \\ v_b &= v_b(x,t) \\ p &= p(x,t) \end{aligned} \tag{2.20}$$

It is seen that none of the quantities in the Reynolds equation depend on y. In other words the Reynolds equation reduces a 2D physical problem to a 1D mathematical problem. This is very advantageous from a computational point of view as well as for analytical treatment of lubrication films. It is noted that the Reynolds equation in a similar way reduces a 3D physical problem to a 2D mathematical problem. Basically the dimension across the fluid film has been eliminated by the integrations (2.7) and (2.8).

The next section is devoted to the relationship between solid body motion and the parameters h_a, h_b, u_a, u_b, v_a, and v_b. After treatment of the general case we provide some special cases, which are often found in applications.

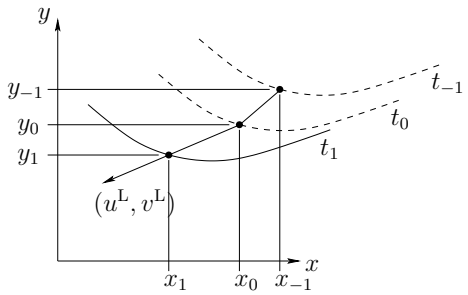

Figure 2.5: A point on a solid body for different instants of time.

2.2 Arbitrary solid body motion

As mentioned in the previous section the Reynolds equation is formulated using Eulerian coordinates. Typically, however, the motion of the solid bodies that are separated by the fluid film are described using Lagrangian coordinates. The difference in representation causes a "difficulty" when relating velocities and positions. Usually one would think that velocity is the time derivative of position. However, this kinematic rule must be interpreted correctly, when dealing with Eulerian coordinates.

In figure 2.5 we have sketched a point on a body as it moves in time. We see that the velocity of the point is given by the (total) derivative of the position of the point with respect to time

$$\begin{aligned} u^{\mathrm{L}} &= \frac{\mathrm{d}x}{\mathrm{d}t} \approx \frac{x_1 - x_0}{t_1 - t_0} \\ v^{\mathrm{L}} &= \frac{\mathrm{d}y}{\mathrm{d}t} \approx \frac{y_1 - y_0}{t_1 - t_0} \end{aligned} \tag{2.21}$$

The superscript $^{\mathrm{L}}$ indicates that the velocity components are obtained from Lagrangian coordinates. Note that the origin of the change in position is immaterial – it may be due to rigid body translation and/or rotation as well as due to the flexibility of the solid.

Now let us translate the Lagrangian coordinates of the solid body a to the Eulerian coordinates of the Reynolds equation (body b is treated the same way). First, it is realized that the Reynolds equation is defined on a 1D domain with x and t being the independent variables. Thus we see that

$$h_a(x,t) = y^{\mathrm{E}}(x,t) \tag{2.22}$$

The notation $y^{\mathrm{E}}(x,t)$ indicates that y must be expressed as a Eulerian coordinate. However, what we have available from the description of the solid body a is a Lagrangian description. The link between these representations can be written

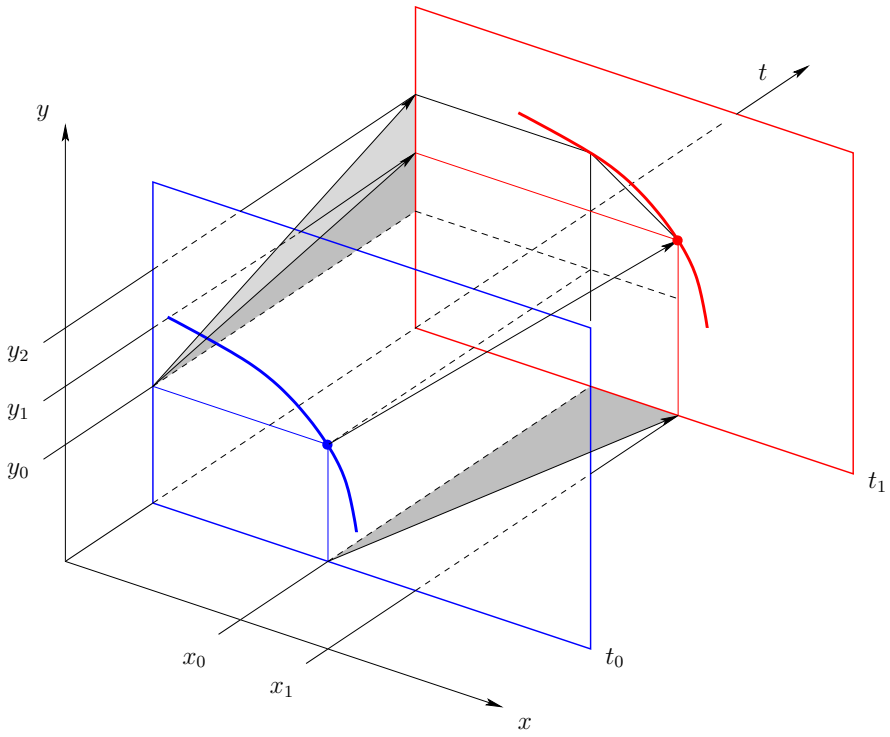

Figure 2.6: This figure shows the position of a solid body (thick line) at two moments in time. Blue means t_0 and red means t_1. It is seen that y measured at x_0 changes from y_0 to y_2 due to *two* mechanisms 1) the body is elevated from y_0 to y_1, 2) the body translates in the x-direction from x_0 to x_1. Refer to the text for the details on what this means for the velocity *felt* at x_0 in the y-direction.

as

$$y^{\mathrm{E}}(x(t),t) = y^{\mathrm{L}}(x_0,t) \tag{2.23}$$

The right hand side says that y is a function of time t, and dependent on the parameter x_0, which denotes the location of the point we are following at time t_0 (for different x_0 we follow different points on the solid body). The left hand side treats y as a field $y = y(x,t)$, and indicates that y^{E} matches y^{L} when x is chosen to coincide with $x(t)$ at the current time.

In order to obtain the velocity components u_a and v_a for the Reynolds equation we differentiate (2.23) with respect to time

$$\frac{\partial y^{\mathrm{E}}}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial y^{\mathrm{E}}}{\partial t} = \frac{\mathrm{d}y^{\mathrm{L}}}{\mathrm{d}t} \tag{2.24}$$

This equation shows that (referring to (2.17) and (2.22))

$$v_a = \frac{\partial h_a}{\partial t} = \frac{\partial y^{\mathrm{E}}}{\partial t} = \frac{\mathrm{d}y^{\mathrm{L}}}{\mathrm{d}t} - \frac{\partial y^{\mathrm{E}}}{\partial x}\frac{\partial x}{\partial t} \tag{2.25}$$

In this equation $\partial y^{\mathrm{E}}/\partial x$ is the slope on the solid body a and $\partial x/\partial t$ is the velocity in the x-direction, both evaluated at the current point of interest.

This result can also be found from geometrical considerations. From figure 2.6 we observe

$$\begin{aligned} u^{\mathrm{L}} &= \frac{\mathrm{d}x}{\mathrm{d}t} \approx \frac{x_1 - x_0}{t_1 - t_0} \\ v^{\mathrm{L}} &= \frac{\mathrm{d}y}{\mathrm{d}t} \approx \frac{y_1 - y_0}{t_1 - t_0} \end{aligned} \tag{2.26}$$

However, the change in y measured at a fixed point $x = x_0$ becomes

$$(y_2 - y_0) = (y_1 - y_0) - \alpha(x_1 - x_0) \tag{2.27}$$

where

$$\alpha = \frac{y_1 - y_2}{x_1 - x_0} \tag{2.28}$$

Expressing this on a time basis gives

$$\frac{y_2 - y_0}{t_1 - t_0} = \frac{y_1 - y_0}{t_1 - t_0} - \alpha\frac{x_1 - x_0}{t_1 - t_0} \tag{2.29}$$

Making $\Delta t = t_1 - t_0$ infinitely small transforms each term into differential quotients. Using (2.26) and the usual notation for partial and total derivatives gives

$$\frac{\partial y}{\partial t} = \frac{\mathrm{d}y}{\mathrm{d}t} - \frac{\partial y}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} = v^{\mathrm{L}} - \frac{\partial y}{\partial x}u^{\mathrm{L}} \tag{2.30}$$

which is the same as (2.25). The formula above expresses the link between the Lagrangian description and the Eulerian description, which is what we were looking for.

The same kind of derivation can be carried out for the solid b, so in total we get

$$\begin{aligned} u_a &= U_a \\ v_a &= V_a - \frac{\partial h_a}{\partial x}U_a \end{aligned} \tag{2.31}$$

and

$$\begin{aligned} u_b &= U_b \\ v_b &= V_b - \frac{\partial h_b}{\partial x}U_b \end{aligned} \tag{2.32}$$

where we have adopted the notation $U = \mathrm{d}x/\mathrm{d}t$ and $V = \mathrm{d}y/\mathrm{d}t$, $h = y^{\mathrm{E}}$ and $\partial h/\partial x = \partial y^{\mathrm{E}}/\partial x$. Thus capital letters U and V represent the Lagrangian velocity components on a solid body. Substituting this into Reynolds equation (2.19) gives

$$\begin{aligned}
&\frac{\partial}{\partial x}\left(\left[\frac{\rho(h_a^3 - h_b^3)}{6\mu} - \frac{\rho(h_a^2 - h_b^2)(h_a + h_b)}{4\mu} + \frac{\rho(h_a - h_b)h_a h_b}{2\mu}\right]\frac{\partial p}{\partial x}\right) \\
&\quad = -\frac{\partial}{\partial x}\left(\frac{\rho(h_a - h_b)(U_a + U_b)}{2}\right) \\
&\qquad + \rho(V_b - \frac{\partial h_b}{\partial x}U_b - V_a + \frac{\partial h_a}{\partial x}U_a) \\
&\qquad - (h_a - h_b)\frac{\partial \rho}{\partial t}
\end{aligned} \tag{2.33}$$

where

x independent variable
t independent variable
$p = p(x,t)$ pressure
$\rho = \rho(x,t)$ density
$\mu = \mu(x,t)$ dynamic viscosity
$h_a = h_a(x,t)$ y-position of upper body
$U_a = U_a(x,t)$ u-velocity on upper body (Lagrangian component)
$V_a = V_a(x,t)$ v-velocity on upper body (Lagrangian component)
$h_b = h_b(x,t)$ y-position of lower body
$U_b = U_b(x,t)$ u-velocity on lower body (Lagrangian component)
$V_b = V_b(x,t)$ v-velocity on lower body (Lagrangian component)

In figure 2.7 the identifiers presented above are shown graphically. As mentioned already Reynolds equation is 1D in the mathematical sense (in space), even though it models a 2D physical situation. We have indicated this in figure 2.7 by separating the axis measuring film thickness and the axis containing the domain for Reynolds equation.

In the following sections we provide special cases of the Reynolds equation. A graphical representation of the special configurations is shown in figure 2.8.

2.3 Special case $h_b = 0$

A common situation is that body b is simply a plane. In that case the coordinate system may be chosen such that $h_b = 0$. Furthermore we have $\partial h_b/\partial x = 0$. The Reynolds Equation reduces to

$$\frac{\partial}{\partial x}\left(\frac{\rho h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\rho h_a(U_a + U_b)}{2}\right) - \rho(V_b - V_a + \frac{\partial h_a}{\partial x}U_a) + h_a\frac{\partial \rho}{\partial t} \tag{2.34}$$

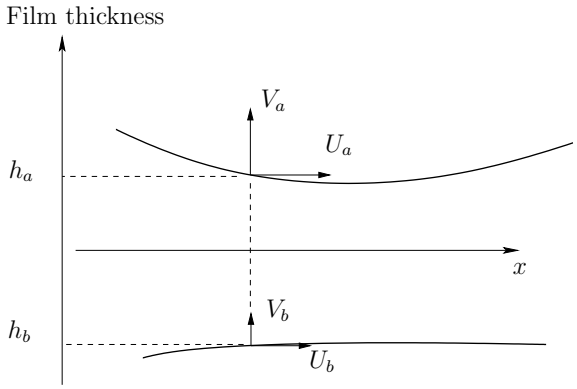

Figure 2.7: A graphical view of the parameters for the Reynolds equation. Reynolds equation is defined along the x-axis. Note that the velocity components and film thicknesses are always to be evaluated at a given x, regardless of how the solid bodies move. This necessitates a transformation from Lagrangian coordinates to Eulerian coordinates as described in section 2.2.

2.4 Special case $h_b = V_b = U_a = 0$

If body b is plane and does not move in the y-direction and body a does not move in the x-direction, then Reynolds Equation may be simplified further

$$\frac{\partial}{\partial x}\left(\frac{\rho h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\rho h_a U_b}{2}\right) + \rho V_a + h_a\frac{\partial \rho}{\partial t} \tag{2.35}$$

This formula is popular for the simulation of piston rings. One would then take the liner (body b) to be moving, while the piston ring (body a) would be fixed in the x-direction. Effects from squeeze action are still included since no assumption on V_a is made.

Note that it becomes easy to evaluate h_a at specified points, because body a does not move relatively to the domain where Reynolds equation is defined. This makes (2.35) a particular attractive formulation.

2.5 Special case $h_b = V_b = V_a = 0$

Another special case is the situation of pure sliding. There is no squeeze action since V_a and V_b are both zero. The Reynolds equation becomes

$$\frac{\partial}{\partial x}\left(\frac{\rho h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\rho h_a (U_a + U_b)}{2}\right) - \rho\frac{\partial h_a}{\partial x}U_a + h_a\frac{\partial \rho}{\partial t} \tag{2.36}$$

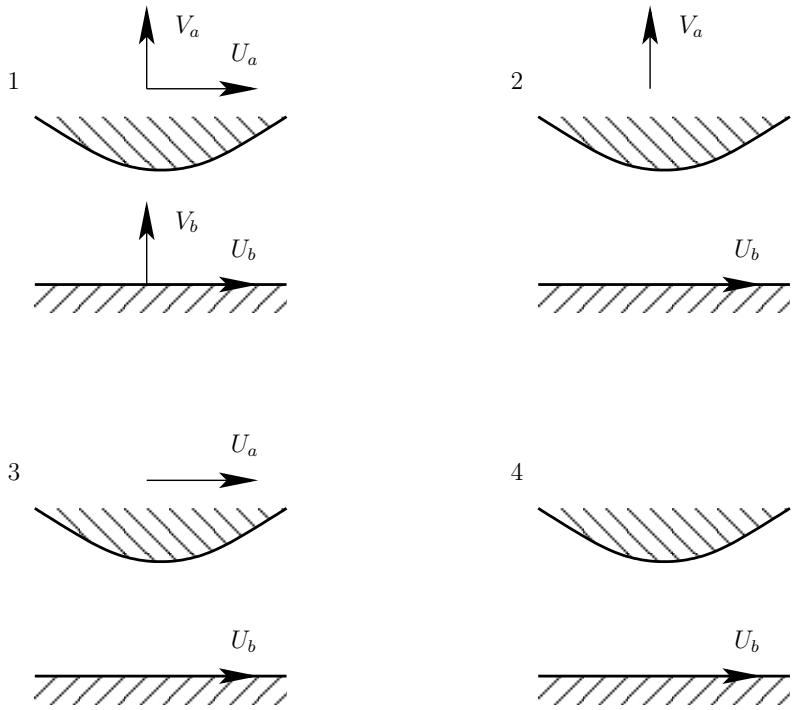

Figure 2.8: Special cases of the Reynolds equation 1) equation (2.34), 2) equation (2.35), 3) equation (2.36) and (2.37), and 4) equation (2.40) and (2.41).

If furthermore the fluid is incompressible so $\rho = \rho_0$ (density cancels out) and the bodies are rigid we get

$$\frac{\partial}{\partial x}\left(\frac{h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{U_b - U_a}{2}\frac{\partial h_a}{\partial x} \tag{2.37}$$

This result was also found in Gohar [26] under the same assumptions. This equation shows that if body a and body b are moving at the same velocity there is no source term for the generation of pressure.

The same result can be obtained directly from (2.36) by imposing a block velocity U_0 on the system

$$\frac{\partial}{\partial x}\left(\frac{\rho h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\rho h_a((U_a - U_0) + (U_b - U_0))}{2}\right) - \rho\frac{\partial h_a}{\partial x}(U_a - U_0) + h_a\frac{\partial \rho}{\partial t} \tag{2.38}$$

If the bodies are rigid then U_a and U_b have constant values. Thus choosing $U_0 = U_a$ turns (2.38) into

$$\frac{\partial}{\partial x}\left(\frac{\rho h_a^3}{12\mu}\frac{\partial p}{\partial x}\right) = \frac{U_b - U_a}{2}\frac{\partial(\rho h_a)}{\partial x} + h_a\frac{\partial \rho}{\partial t} \tag{2.39}$$

which has the same form as (2.37), but without the assumption of constant density. This transformation is also described in Gohar [26].

2.6 Special case $h_b = V_b = U_a = V_a = 0$, $\mu = \mu_0$

Two specializations are relevant for this project. They differ by assuming incompressibility or compressibility of the fluid and share the following assumptions

1. viscosity is constant
2. the solids are rigid
3. the upper solid is stationary $U_a = 0$, $V_a = 0$
4. the lower solid is planar $h_b = 0$
5. the lower solid has no vertical movement $V_b = 0$
6. only steady state solutions are considered

These assumptions together with compressibility turn (2.33) into

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\rho h_a^3\frac{\mathrm{d}p}{\mathrm{d}x}\right) = 6\mu U_b\frac{\mathrm{d}(\rho h_a)}{\mathrm{d}x} \tag{2.40}$$

Note that we have exchanged the partial derivative operator ∂ with d because we no longer consider time-dependency (x is the only independent variable).

2.7 Special case $h_b = V_b = U_a = V_a = 0$, $\mu = \mu_0$, $\rho = \rho_0$

The second special case of importance for this project is similar to the previous one, except that now we also assume incompressibility of the fluid. This turns the Reynolds equation into

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(h_a^3\frac{\mathrm{d}p}{\mathrm{d}x}\right) = 6\mu U_b\frac{\mathrm{d}h_a}{\mathrm{d}x} \tag{2.41}$$

Equations (2.40) and (2.41) are the forms of the Reynolds equation relevant for this work.

A final assumption (for the incompressible case as well as the compressible case) is that the upper part is convex. This implies no limitation when considering piston rings, which typically have a barrel-shaped or tapered profile[1]. This assumption is only relevant for the simulation of cavitation (section 4.5).

In the following sections we discuss different methods for solving the Reynolds equation, and we present some useful expressions for post processing.

2.8 Obtaining pressure distribution

The primary variable of interest in relation to the Reynolds equation is the pressure distribution. The solution methods available for the Reynolds equation are determined by mainly two factors 1) the fluid properties, and 2) the number of spacial dimensions.

If the fluid is assumed to be compressible and/or is having variable viscosity, then Reynolds equation becomes more complicated. For example if a pressure-density relation and/or a pressure-viscosity relation is introduced the equation becomes nonlinear. In this case analytical solutions will in general not be available.

The number of spacial dimensions is also very important. If only two dimensional flow and steady state is considered Reynolds equation takes the form of an ordinary differential equation. If the fluid properties (density and viscosity) are assumed to be constant, the equation can be solved analytically for geometries of e.g. any polynomial shape. However, the expressions can become lengthy for other than low order polynomials. If a three dimensional flow is considered the Reynolds equation becomes a partial differential equation, even at steady state. In this case analytical results exist only for a few special cases. An example is the parallel step bearing for incompressible fluid with constant viscosity. For 3D simulations it is in general necessary to solve the Reynolds equation numerically.

[1] If the flexibility of the piston ring is taken into account the profile might no longer be strictly convex. However, in this work all solid bodies are considered rigid.

In this thesis we consider only 2D flow. When dealing with the Reynolds equation we will assume steady conditions, constant viscosity, and in most cases also incompressibility. In these cases the Reynolds equation can actually be solved analytically. However, we choose to solve it numerically in all circumstances. This will be the subject for chapter 4.

2.9 Obtaining flow rates

Once the pressure distribution is known it is possible to calculate the volume flow rate through a section at any point where the Reynolds equation is defined. The following formula is found from (2.15) while substituting the previously mentioned restrictions $u_a = U_a = 0$ and $h_b = 0$

$$q = \frac{m}{\rho} = -\frac{h_a^3}{12\mu}\frac{\mathrm{d}p}{\mathrm{d}x} + \frac{u_b h_a}{2} \tag{2.42}$$

The derivative $\mathrm{d}p/\mathrm{d}x$ may be found from p by analytical or numerical differentiation depending on the solution method used for p.

2.10 Obtaining velocity distributions

If the fluid is assumed to be incompressible the velocity field must be divergence free. This means that the following equation must hold at any point in the fluid domain

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{2.43}$$

An expression for the x-component of velocity (2.13) was found during the derivation of the Reynolds equation. It is restated here using the restrictions $u_a = U_a = 0$ and $h_b = 0$

$$u(x,y) = -y\left(\frac{h_a - y}{2\mu}\right)\frac{\mathrm{d}p}{\mathrm{d}x} + u_b\left(\frac{h_a - y}{h_a}\right) \tag{2.44}$$

We can now rearrange (2.43) in order to get an ordinary differential equation for v

$$\begin{aligned}\frac{\partial v}{\partial y} &= -\frac{\partial u}{\partial x} \\ &= \frac{1}{2\mu}\left[\left(\frac{\mathrm{d}h_a}{\mathrm{d}x}\frac{\mathrm{d}p}{\mathrm{d}x} + h_a\frac{\mathrm{d}^2p}{\mathrm{d}x^2}\right)y - \frac{\mathrm{d}^2p}{\mathrm{d}x^2}y^2\right] - \frac{u_b}{h_a^2}\frac{\mathrm{d}h_a}{\mathrm{d}x}y\end{aligned} \tag{2.45}$$

Integrating from $\eta = 0$ to $\eta = y$ gives, since $v(x,0) = v_b = V_b = 0$

$$v(x,y) = \frac{1}{4\mu}\left[\left(\frac{\mathrm{d}h_a}{\mathrm{d}x}\frac{\mathrm{d}p}{\mathrm{d}x} + h_a\frac{\mathrm{d}^2p}{\mathrm{d}x^2}\right)y^2 - \frac{2}{3}\frac{\mathrm{d}^2p}{\mathrm{d}x^2}y^3\right] - \frac{u_b}{2h_a^2}\frac{\mathrm{d}h}{\mathrm{d}x}y^2 \tag{2.46}$$

which is an expression for v evaluated at any point (x, y). As in the previous section one must obtain the various derivatives either analytically or by numerical differentiation depending on whether a continuous or discrete solution for p is available.

Chapter 3

Boundary conditions for the Reynolds equation

In this chapter we present the boundary conditions needed for the Reynolds equation. Along the way we show plots of the resulting pressure distribution. This is done in order to clarify what is going on. How to actually calculate the pressure distribution is the subject of the next chapter.

The general Reynolds equation is a partial differential equation defined in one or two spacial dimensions and possibly evolving in time. The special cases of our interest (2.40) and (2.41) are second order ordinary differential equations.

Two boundary conditions are needed in order to get unique solutions. A straight forward approach is to specify the value of pressure at the inlet and at the outlet point of the lubricated conjunction. One can operate with either absolute pressure, gauge pressure, or a non-dimensionalized pressure. In figure 3.1 we show the resulting pressure distribution for a fixed incline slider bearing with the following properties

$$\begin{aligned} L_{\text{fi}} &= 10 \ \text{ length of bearing, mm} \\ p_{\text{in}} &= 101500 \ \text{ inlet pressure, Pa} \\ p_{\text{out}} &= 101500 \ \text{ outlet pressure, Pa} \\ U &= 5 \ \text{ horizontal velocity of lower part, m/s} \\ s_h &= 20 \ \text{ inclination, } \mu\text{m} \\ h_{\min} &= 10 \ \text{ minimum film thickness, } \mu\text{m} \\ h_{\text{in}} &= h_{\min} + s_h = 30 \ \text{ height at inlet, } \mu\text{m} \\ h_{\text{out}} &= h_{\min} = 10 \ \text{ height at outlet, } \mu\text{m} \\ \mu_0 &= 0.05 \ \text{ dynamic viscosity, Pa s} \\ \rho &\ \text{ no value needed for incompressible fluid, kg/m}^3 \end{aligned} \tag{3.1}$$

The upper part is stationary. It is seen that the pressure becomes positive everywhere on the domain where the Reynolds equation has been applied. In

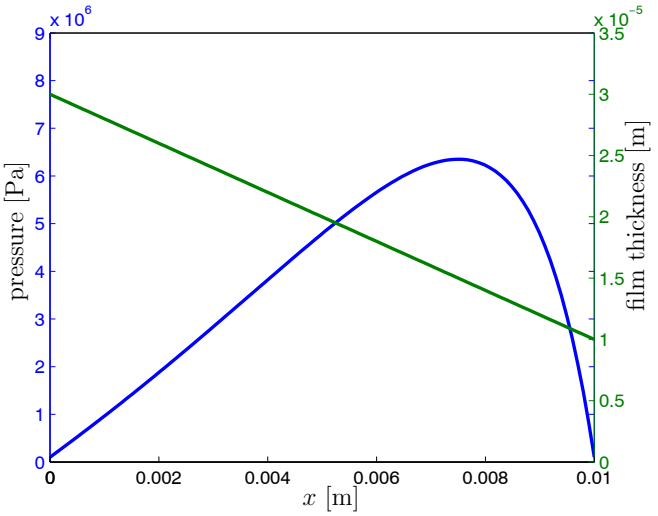

Figure 3.1: Boundary conditions and pressure distribution for the fixed incline slider bearing (3.1). Blue: Pressure level, Green: Oil film thickness.

this case direct specification of the pressure at the end points gives physically meaningful results.

However, if for some reason pressure should become negative, then special care is needed. This is due to the fact that liquids (approximately) cannot sustain tensile force. Instead the liquid will experience *cavitation*, which is the subject of the next section.

3.1 Cavitation

Cavitation is a phenomenon that can be observed in low pressure regions of liquids. Cavitation is a combination of two mechanisms 1) dissolved air bubbles will grow (expand), 2) instantaneous evaporation of the liquid takes place. The effect of cavitation is that pressure in reality cannot get lower than the equilibrium evaporation pressure of the fluid. We denote this pressure level by p_{cav}. The cavitation pressure is more or less equal to the ambient pressure, and it is customary to set $p_{\text{cav}} = p_{\text{amb}}$ when using the Reynolds equation. It is noted though, that liquids in reality are able to sustain small tensile forces for a short moment of time. These effects are neglected in the following treatment of cavitation.

The source of cavitation is low pressure, which in relation to lubrication has two main origins 1) diverging geometries (relative to sliding motion), 2) squeeze action (separation of the lubricated bodies). In this thesis we do not consider

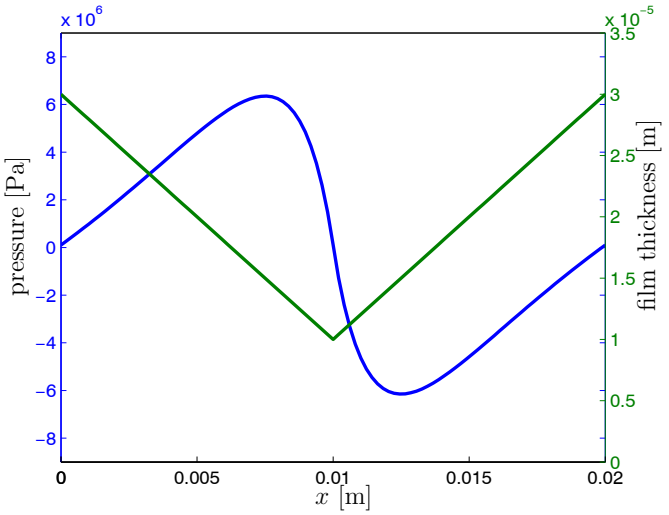

Figure 3.2: Full Sommerfeld solution for the converging-diverging geometry (3.2).

squeeze action, but the treatment of cavitation remains valid even if squeeze action is included. Cavitation appears for low pressure, irrelevant of what caused the low pressure.

Consider the converging-diverging geometry shown in figure 3.2. We have reused all the parameters from (3.1) except for the geometry of the upper part of the bearing. A symmetrical part has been added so that the geometry is now given by

$$h = \begin{cases} s_h \dfrac{L_{\text{fi}} - x}{L_{\text{fi}}} + h_{\min}, & 0 \leq x < L_{\text{fi}} \\ s_h \dfrac{x - L_{\text{fi}}}{L_{\text{fi}}} + h_{\min}, & L_{\text{fi}} \leq x \leq L_{\text{cd}} \end{cases} \tag{3.2}$$

where

L_{fi} length of fixed incline slider bearing (3.1)
$L_{\text{cd}} = 2L_{\text{fi}}$ length of converging-diverging bearing

Again we have plotted the resulting pressure distribution, when using the boundary conditions of ambient pressure at the inlet and outlet point. It is seen that positive pressure is present on the converging part of the lubricated conjunction, while pressure below ambient pressure is present on the diverging part. This kind of solution, which is taken "as is" is known as the Full Sommerfeld solution. Clearly, the part with negative pressure does not make sense, since absolute pressure in reality can never become negative.

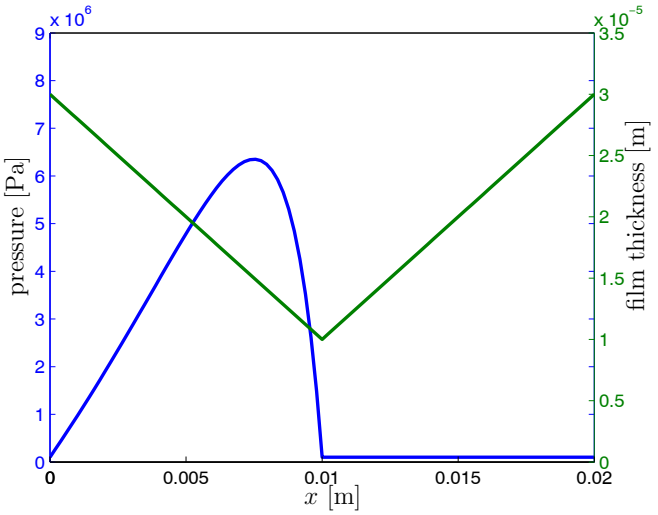

Figure 3.3: Half Sommerfeld solution for the converging-diverging geometry (3.2).

A first rough approximation of the true pressure distribution, when cavitation is present, is known as the Half Sommerfeld solution. Here pressure below cavitation pressure is simply changed manually into the cavitation pressure. This solution for the converging-diverging geometry is shown in figure 3.3. Although attractive because of its simplicity we do not use the Half Sommerfeld solution in practice. It can be shown that the mass flow entering the conjunction at $x = 0$ is not conserved at the outlet $x = L_{\text{cd}}$ [24].

As a replacement for the Sommerfeld solutions different *cavitation criteria* have been proposed. The most well-known is probably the Reynolds cavitation criteria. It states that

$$\begin{aligned} p(x_{\text{cav}}) &= p_{\text{cav}} \\ \frac{\mathrm{d}p(x_{\text{cav}})}{\mathrm{d}x} &= 0 \end{aligned} \tag{3.3}$$

where x_{cav} denotes the location of the onset of cavitation. In figure 3.4 we plot the pressure distribution for the converging-diverging geometry using the Reynolds cavitation criteria. The inlet boundary condition is $p(x_{\text{in}}) = p_{\text{amb}}$ as before. The outlet boundary condition is given by (3.3). This gives three boundary conditions in total. However, one extra degree of freedom is added, namely the a priori unknown location of the onset of cavitation x_{cav}. The domain of definition for the Reynolds equation is in other words not $x \in [0, L_{\text{cd}}]$, but instead $x \in [0, x_{\text{cav}}]$. It should be clear that h_a and/or h_b in the Reynolds equation becomes a function of x_{cav}. Therefore the Reynolds equation becomes nonlinear, and x_{cav} must be

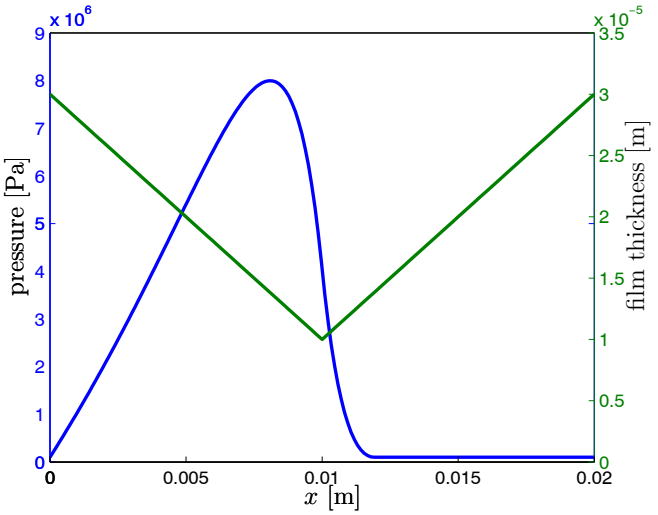

Figure 3.4: Pressure distribution of the converging-diverging geometry (3.2) using the Reynolds cavitation criteria.

found by iteration. The solution for pressure will satisfy one condition at the inlet (the specified pressure) and two conditions at the onset of cavitation (the specified cavitation pressure and zero slope of the pressure curve).

Finally we need to consider the possibility of rebuild-up of the pressure curve. This situation can occur if the pressure at the outlet is bigger than the cavitation pressure or if squeeze action takes place. Figure 3.5 shows an example of this for $p_{\text{out}} = 5.1$ MPa. In this case we see that it is actually necessary to apply the Reynolds equation two times on two distinct subdomains. The first domain is defined by $x \in [0, x_{\text{cav}}]$, and the second domain is defined by $x \in [x_{\text{rebuildup}}, L_{\text{cd}}]$. The value of $x_{\text{rebuildup}}$ is found by calculating the flow rate q_{cav} at x_{cav}. The flow rate evaluated at $x_{\text{rebuildup}}$ must match q_{cav}.

It is interesting to compare the various solutions with each other. First it is seen that the pressure distribution on the domain $x \in [0, L_{\text{fi}}]$ is exactly the same on figure 3.1, 3.2, and 3.3. This is due to the symmetrical properties of the converging-diverging geometry compared with the simple fixed incline slider bearing. The same observation can be made for figures 3.4 and 3.5 – the pressure distribution at $x \in [0, x_{\text{cav}}]$ is the same. However, there is a difference if we compare the Sommerfeld solutions and the solutions with the Reynolds cavitation criteria. It is seen that the maximum pressure is significantly larger, when using the Reynolds cavitation criteria.

This ends the section on boundary conditions for the Reynolds equation. It

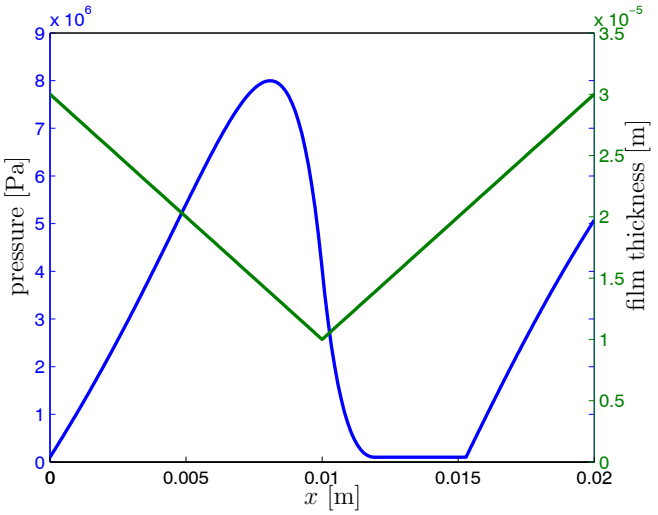

Figure 3.5: An example of rebuild-up of pressure due to the outlet pressure level.

should be noted that other cavitation assumptions than the Reynolds cavitation criteria exist (having names such as *open end*, *free end*, and others). See for example [18] for more information. How to iterate a solution for the Reynolds equation when cavitation is present is discussed in the next chapter.

Chapter 4

Discretization of the Reynolds equation

In this chapter we present a discretization method for the Reynolds equation. It should be noted that we do not consider the Reynolds equation in its most general form. Only the special cases (2.40) and (2.41) are handled. This chapter also includes a section on sensitivity analysis of the Reynolds equation, and a section on cavitation iteration.

4.1 Incompressible case

In this section we provide a discretization method for (2.41), which we restate below for convenience

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(h^3\frac{\mathrm{d}p}{\mathrm{d}x}\right) = 6\mu U\frac{\mathrm{d}h}{\mathrm{d}x} \tag{4.1}$$

For notational convenience we have dropped the subscript $_a$ on h and the subscript $_b$ on U as compared with (2.41). We see immediately that (4.1) could be integrated once with respect to x, since viscosity μ and velocity U of the lower part is constant. This would result in a first order differential equation involving an unknown constant of integration C. It is possible to discretize the resulting equation and solve it simultaneously for pressure p and the integration constant C. However, we find this method less straight forward and the system matrix will not be banded.

Instead of the procedure outlined above, we keep (4.1) as a second order differential equation. We expand the nested differentiation as follows

$$\frac{\mathrm{d}(h^3)}{\mathrm{d}x}\frac{\mathrm{d}p}{\mathrm{d}x} + h^3\frac{\mathrm{d}^2p}{\mathrm{d}x^2} = 6\mu U\frac{\mathrm{d}h}{\mathrm{d}x} \tag{4.2}$$

We approximate the derivatives by finite differences. For evenly distributed nodes

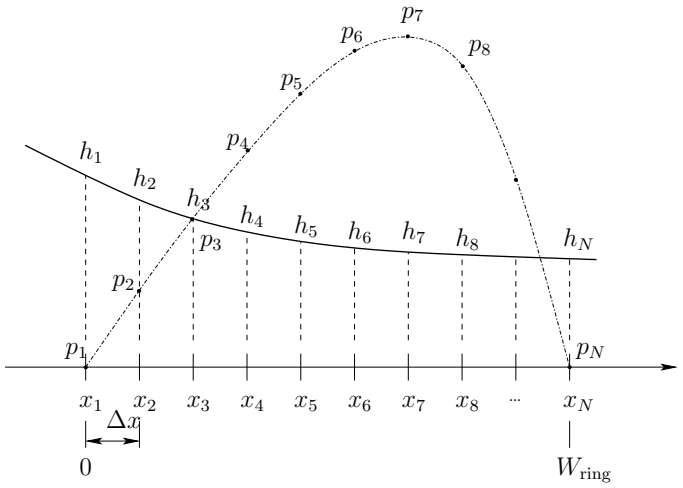

Figure 4.1: Variables used for finite difference discretization of the Reynolds equation.

the following formulas can be derived [21]

$$\begin{aligned}\frac{\mathrm{d}p}{\mathrm{d}x}|_{x=x_i} &= \frac{p_{i+1}-p_{i-1}}{2(\Delta x)} + O\left((\Delta x)^2\right)\\ \frac{\mathrm{d}^2p}{\mathrm{d}x^2}|_{x=x_i} &= \frac{p_{i+1}-2p_i+p_{i-1}}{(\Delta x)^2} + O\left((\Delta x)^2\right)\end{aligned} \tag{4.3}$$

where Δx is the nodal spacing given by

$$\Delta x = \frac{L}{N-1} \tag{4.4}$$

In this expression L denotes the length of the lubricated domain and N is the number of discretization points. The expressions in (4.3) are second order accurate. Substituting (4.3), while neglecting higher order terms, into (4.2) gives

$$\frac{(h_{i+1})^3-(h_{i-1})^3}{2(\Delta x)}\frac{p_{i+1}-p_{i-1}}{2(\Delta x)} + (h_i)^3\frac{p_{i+1}-2p_i+p_{i-1}}{(\Delta x)^2} = 6\mu U\frac{h_{i+1}-h_{i-1}}{2(\Delta x)} \tag{4.5}$$

valid for $i = 2, 3, 4, \ldots, N-1$. In figure 4.1 a graphical representation of the discrete variables is shown. Collecting terms with p gives the following useful matrix-vector form

$$\begin{bmatrix} -(h_{i+1})^3+(h_{i-1})^3+4(h_i)^3 \\ -8(h_i)^3 \\ (h_{i+1})^3-(h_{i-1})^3+4(h_i)^3 \end{bmatrix}^{\mathrm{T}} \begin{pmatrix} p_{i-1} \\ p_i \\ p_{i+1} \end{pmatrix} = 12(\Delta x)\mu U\left(h_{i+1}-h_{i-1}\right) \tag{4.6}$$

valid for $i = 2, 3, 4, \ldots, N-1$. The boundary conditions are enforced via node 1 and node N

$$\begin{aligned} [1]\{p_1\} &= p_{\text{in}} \\ [1]\{p_N\} &= p_{\text{out}} \end{aligned} \tag{4.7}$$

The treatment of the boundary conditions for cavitation is postponed until section 4.4.

Equations (4.6) and (4.7) are assembled into the matrix equation

$$\mathbf{M}\mathbf{p} = \mathbf{b} \tag{4.8}$$

where

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & \ldots \\ -h_3^3 + h_1^3 + 4h_2^3 & -8h_2^3 & h_3^3 - h_1^3 + 4h_2^3 & 0 & \ldots \\ 0 & -h_4^3 + h_2^3 + 4h_3^3 & -8h_3^3 & h_4^3 - h_2^3 + 4h_3^3 & \ldots \\ \vdots & & \ddots & & \ldots \end{bmatrix} \tag{4.9}$$

$$\mathbf{b} = \begin{pmatrix} p_{\text{in}} \\ 12(\Delta x)U(h_3 - h_1) \\ 12(\Delta x)U(h_4 - h_2) \\ \vdots \end{pmatrix} \tag{4.10}$$

This is a tridiagonal system which can be solved efficiently using the Thomas algorithm (complexity is $O(N)$ with N being the number of unknowns).

4.2 Compressible case

In this section we provide a discretization method for the compressible Reynolds equation (2.40), restated here without subscript $_a$ on h and subscript $_b$ on U

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\rho h^3 \frac{\mathrm{d}p}{\mathrm{d}x}\right) = 6\mu U \frac{\mathrm{d}(\rho h)}{\mathrm{d}x} \tag{4.11}$$

Because of compressibility (4.11) is a nonlinear equation. The pressure p and the density ρ are linked through an equation of state

$$p = p(\rho) \quad \text{or} \quad \rho = \rho(p) \tag{4.12}$$

At this point we will not go into what is an appropriate equation of state. We will simply assume that it is available (see section 6.2).

In general nonlinear systems must be solved by iteration. Typically there are several ways of performing the iterations. A basic method is the so-called *functional iteration*[1]. The idea is to fix some variables at the current iteration level,

[1]Functional iteration is also known as *zeta iteration* or *fixed point iteration*.

so that the system can be easily solved, that is, becomes linear in the remaining variables. Then the previously fixed variables are updated using the values of the recently calculated variables. This procedure is repeated until convergence.

We will apply functional iteration in the following way. Solve

$$\frac{\mathrm{d}(\rho^k h^3)}{\mathrm{d}x}\frac{\mathrm{d}p^{k+1}}{\mathrm{d}x} + \rho^k h^3 \frac{\mathrm{d}^2 p^{k+1}}{\mathrm{d}x^2} = 6\mu U \frac{\mathrm{d}(\rho^k h)}{\mathrm{d}x} \tag{4.13}$$

for p^{k+1} while ρ^k is constant and k denotes iteration level. Then update

$$\rho^{k+1} = \begin{cases} \rho(p^{k+1}) & p^{k+1} > 0 \\ \rho(0) & p^{k+1} \leq 0 \end{cases} \tag{4.14}$$

using the equation of state (Branching is needed in case cavitation takes place. If the equation of state is not prepared for cavitation, the pressure might become negative, and evaluation of $\rho = \rho(p)$ could be without meaning.) Repeat this procedure until convergence. In the following (4.11) is discretized without showing the iteration superscript k.

The discretization of (4.11) follows the same steps as for the incompressible case. Again we begin by expanding the nested differentiation on the left hand side

$$\frac{\mathrm{d}(\rho h^3)}{\mathrm{d}x}\frac{\mathrm{d}p}{\mathrm{d}x} + \rho h^3 \frac{\mathrm{d}^2 p}{\mathrm{d}x^2} = 6\mu U \frac{\mathrm{d}(\rho h)}{\mathrm{d}x} \tag{4.15}$$

The next manipulations are similar to the incompressible case, so we state the final expressions directly

$$\begin{aligned} &\begin{bmatrix} -\rho_{i+1}(h_{i+1})^3 + \rho_{i-1}(h_{i-1})^3 + 4\rho_i(h_i)^3 \\ -8\rho_i(h_i)^3 \\ \rho_{i+1}(h_{i+1})^3 - \rho_{i-1}(h_{i-1})^3 + 4\rho_i(h_i)^3 \end{bmatrix}^{\mathrm{T}} \begin{pmatrix} p_{i-1} \\ p_i \\ p_{i+1} \end{pmatrix} \\ &= 12(\Delta x)\mu U \left(\rho_{i+1}h_{i+1} - \rho_{i-1}h_{i-1}\right) \end{aligned} \tag{4.16}$$

valid for $i = 2, 3, 4, \ldots, N-1$. The boundary conditions are enforced via node 1 and node N

$$\begin{aligned} [1]\{p_1\} &= p_{\mathrm{in}} \\ [1]\{p_N\} &= p_{\mathrm{out}} \end{aligned} \tag{4.17}$$

4.3 Sensitivity analysis

In this section we wish to perform a sensitivity analysis of the Reynolds equation with respect to the inlet point x_{in}. There are basically two methods available 1) analysis by perturbations, and 2) differentiation of the system equation.

The first method is based on perturbation of the variable for which we want to know the sensitivity of p. For example we could be looking for the sensitivity with respect to h. One would then substitute

$$\begin{aligned} h &= h_0 + \Delta h \\ p &= p_0 + p_h \Delta h \end{aligned} \tag{4.18}$$

into the Reynolds equation, and then collect terms with h_0 to build an equation for the equilibrium solution (this becomes the Reynolds equation). Then collecting terms with Δh would result in an equation for the sensitivity of p on h, which could be discretized by the methods of the previous section. This method is described in detail in [28].

The second method is applied directly to the discrete equations written as a matrix equation

$$\mathbf{M}\mathbf{p} = \mathbf{b} \tag{4.19}$$

This equation could for example represent equation (4.8). Differentiation with respect to some variable of interest gives

$$\mathbf{M}'\mathbf{p} + \mathbf{M}\mathbf{p}' = \mathbf{b}' \tag{4.20}$$

If $\mathbf{p}$ is already known from the solution of (4.19) we can rearrange the previous equation in the following way

$$\mathbf{M}\mathbf{p}' = \mathbf{b}' - \mathbf{M}'\mathbf{p} \tag{4.21}$$

The solution $\mathbf{p}'$ is precisely the sensitivity of p (defined in discrete points) with respect to some variable of interest. We will use this method to obtain the sensitivity on p with respect to the location of the inlet point x_{in}. We will need these sensitivities in the next chapter.

The system we consider is the incompressible Reynolds equation as expressed in (4.8). For interior nodes $i = 2, 3, \ldots, N-1$ we get for the derivative of the right hand side vector $\mathbf{b}$ (4.10)

$$\frac{\partial b_i}{\partial x_{\text{in}}} = 12\mu U \left[\frac{\partial \Delta x}{\partial x_{\text{in}}} (h_{i+1} - h_{i-1}) + \Delta x \frac{\partial (h_{i+1} - h_{i-1})}{\partial x_{\text{in}}} \right] \tag{4.22}$$

The expression above has three undefined symbols. The first $\partial \Delta x / \partial x_{\text{in}}$ is a measure of how Δx changes when x_{in} is changed. The layout of the discretization points is given by

$$x_i = x_{\text{in}} + \Delta x (i-1) = \frac{N-i}{N-1} x_{\text{in}} + \frac{i-1}{N-1} x_{\text{out}} \tag{4.23}$$

where

$$\Delta x = \frac{x_{\text{out}} - x_{\text{in}}}{N-1} \tag{4.24}$$

From this we see that

$$\frac{\partial \Delta x}{\partial x_{\text{in}}} = \frac{-1}{N-1} \tag{4.25}$$

The remaining undefined symbols in (4.22) are $\partial h_{i+1}/\partial x_{\text{in}}$ and $\partial h_{i+1}/\partial x_{\text{in}}$. Instead of treating them independently we provide below an expression for $\partial h_i/\partial x_{\text{in}}$. It is then a simple matter to increment or decrement the index i as needed. Using the chain rule gives

$$\frac{\partial h_i}{\partial x_{\text{in}}} = \frac{\partial h_i}{\partial x_i}\frac{\partial x_i}{\partial x_{\text{in}}} \approx \frac{h_{i+1} - h_{i-1}}{2\Delta x}\frac{N-i}{N-1} \tag{4.26}$$

Here we have used (4.23) to obtain an expression for $\partial x_i/\partial x_{\text{in}}$.

Taking a look at (4.22) reveals that we need to evaluate (4.26) for $i = 1, 2, \ldots, N$. Clearly the endpoints are problematic because the index on h in (4.26) will come out of bounds. However, we see that for $i = N$ a factor becomes zero so

$$\frac{\partial h_N}{\partial x_{\text{in}}} = 0 \tag{4.27}$$

For $i = 1$ we resolve the problem by exchanging the central difference scheme with a one-sided finite difference scheme

$$\frac{\partial h_1}{\partial x_{\text{in}}} \approx \frac{-3h_1 + 4h_2 - h_3}{2\Delta x} \tag{4.28}$$

Summarizing we get the follow expression for $\mathbf{b}'$

$$\begin{aligned}
b'_1 &= 0 \\
b'_2 &= 12\mu U \left[-\frac{1}{N-1}(h_3 - h_1) \right. \\
&\quad \left. + \Delta x \left(\frac{h_4 - h_2}{2\Delta x}\frac{N-3}{N-1} - \frac{-3h_1 + 4h_2 - h_3}{2\Delta x} \right) \right] \\
b'_i &= 12\mu U \left[-\frac{1}{N-1}(h_{i+1} - h_{i-1}) \right. \\
&\quad \left. + \Delta x \left(\frac{h_{i+2} - h_i}{2\Delta x}\frac{N-(i+1)}{N-1} - \frac{h_i - h_{i-2}}{2\Delta x}\frac{N-(i-1)}{N-1} \right) \right] \\
b'_{N-1} &= 0 \\
b'_N &= 0
\end{aligned} \tag{4.29}$$

where $i = 3, 4, \ldots, N-2$. The first and the last entry in $\mathbf{b}'$ comes from the boundary conditions (4.7), and are obviously zero.

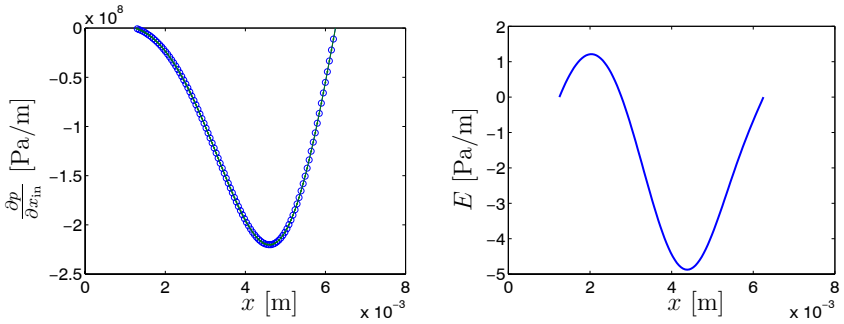

Figure 4.2: Sensitivity of p with respect to x_{in} for a fixed incline slider bearing. Left: '−o−' analytical, '− · −' finite difference. Right: Difference between analytical method and finite difference method.

The derivative of $\mathbf{M}$ with respect to x_{in} is obtained from (4.9). It is seen that we only need to establish one term, namely

$$\frac{\partial (h_i)^3}{\partial x_{\text{in}}} = 3(h_i)^2 \frac{\partial h_i}{\partial x_{\text{in}}} \tag{4.30}$$

in order to build $\mathbf{M}'$. We will not give the details, since an expression for $\partial h_i / \partial x_{\text{in}}$ is already given in (4.26)-(4.28). Note that the first and the last row of $\mathbf{M}'$ has only zeros, because of the boundary conditions (4.7).

In figure 4.2 we have plotted the sensitivity of p with respect to x_{in} and compared it with sensitivities obtained by a finite difference approximation

$$\frac{\partial p_i}{\partial x_{\text{in}}} = \frac{p_i^* - p_i^{**}}{2\delta x}$$

where $\mathbf{p}^*$ is obtained from (4.8) with $x_1 = x_{\text{in}} + \delta x$ and $\mathbf{p}^{**}$ is obtained in a similar way with $x_1 = x_{\text{in}} - \delta x$. The test case used is the fixed incline slider bearing. Note that for this particular case for finite difference approximations in (4.26) and (4.28) are exact. It is seen that the results match very well (agreement to 8 orders of magnitude). This ends the section on sensitivity analysis of the Reynolds equation. Applications are found in section 4.4.4 and 5.3.

4.4 Cavitation iteration – general description

This section is concerned with modeling of cavitation in connection with a numerical solution for the Reynolds equation. In section 3.1 we saw that determination of the point of onset of cavitation is a nonlinear problem, that must be solved by iteration. In this section we discuss briefly five different methods.

1. Equation of state
2. Internal boundary conditions
3. Greedy iterations
4. Newton iterations
5. Bisection method

The following computational examples are based on the converging-diverging geometry already encountered in section 3.1.

4.4.1 Equation of state

The first method is based on a compressible formulation of the Reynolds equation. The idea is to model cavitation by selecting an appropriate equation of state. This equation of state should include a model of the pressure-density relation of a liquid-gas mixture. Using a compressible formulation makes Reynolds equation nonlinear, which necessitates iteration for its solution. No special treatment of the inlet x_{in} or the outlet point x_{out} is needed. We have not investigated this method.

4.4.2 Internal boundary conditions

The second method is based on the use of *internal boundary conditions*. The idea is that, if pressure becomes smaller than the cavitation pressure, then an internal boundary condition is added to the system. In this way the low pressure region is eliminated by enforcing pressure at relevant points to the cavitation pressure. In figure 4.3 we show how the cavitation iteration can be realized. First the Reynolds equation is solved on the entire domain $x \in [0, L_{\text{cd}}]$. Then the node with the smallest value of pressure is detected. If this pressure is smaller than the cavitation pressure, an extra internal boundary condition is added. Referring to figure 4.3 the following modification of the system matrix and right hand side is carried out after the initial solution

$$\begin{aligned} M_{ij} &= 0 \quad \text{for} \quad i = 62, \quad j = 1, 2, \ldots, N, \quad j \neq i \\ M_{ij} &= 1 \quad \text{for} \quad i = 62, \quad j = 62 \\ b_i &= p_{\text{cav}} \quad \text{for} \quad i = 62 \end{aligned} \tag{4.31}$$

where

$\mathbf{M}$ system matrix as given in (4.9)
$\mathbf{b}$ right hand side as given in (4.10)
$N = 99$ number of discretization points

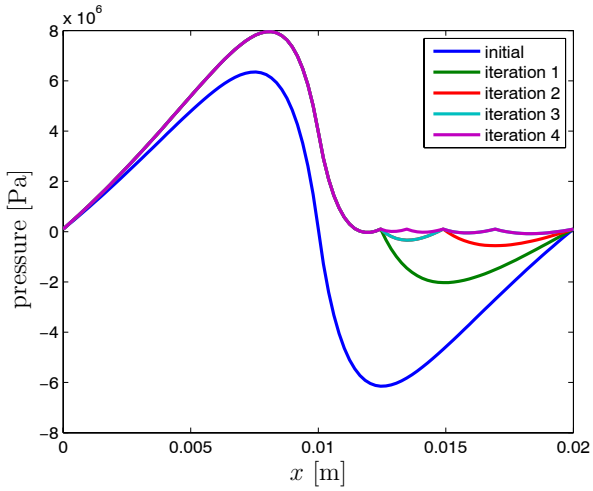

Figure 4.3: Cavitation iteration method "Internal boundary conditions". More iterations than showed here are needed to get the final solution.

The system is solved again and the iteration is repeated until no pressure is smaller than p_{cav}. The drawbacks of this method are that 1) it is slow in its basic form, 2) the onset point of cavitation might not coincide with a discretization point. The first drawback can be alleviated by introducing the internal boundary conditions for more nodes at a time. If multiple cavitation zones exist some book keeping is necessary, which makes the method more complicated. The second drawback is more severe since the onset of cavitation might not coincide with a discretization point. In this case the Reynolds cavitation criteria will not be satisfied exactly. The onset location of cavitation found may be on either side of the true location.

4.4.3 Greedy iterations

The third method is based on modification of the outlet point x_{out}, that is, we iterate the location of x_{out} so that it approaches the point of onset of cavitation x_{cav}. The domain $[0, x_{\text{out}}]$ is remeshed (rediscretized) in each iteration.

The method is less straight forward to implement compared to method 4.4.1 or 4.4.2, because the original domain must be cut in pieces according to the number of cavitation zones. Also the zones of cavitation will not be discretized, which makes post processing less convenient. However, the method has an advantage with respect to efficiency of the iterations. The idea is to first solve the Reynolds equation on the domain $x \in [0, L_{\text{cd}}]$. Then the node with the smallest pressure is detected, say $p_{\min} = p_i$. If $p_{\min} < p_{\text{cav}}$ then the next iteration is prepared by

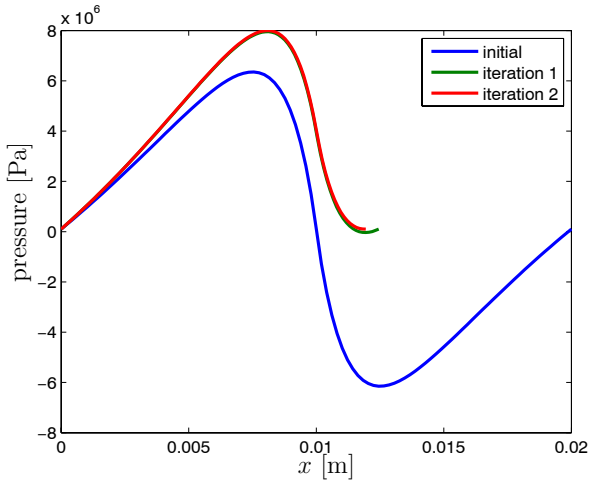

Figure 4.4: Cavitation method "Greedy". Only two iterations are needed for this problem.

setting $x_{\text{out}} = x_i$. This kind of iteration is repeated until no change in x_{out} is detected.

In figure 4.4 the iterations for the converging-diverging test case is shown. As in method 4.4.2 the true onset of cavitation might not be located precisely, but with the greedy iteration method accuracy will be improved if the number of discretization points is kept constant (the distance between discretization points becomes smaller as x_{out} is iterated).

We have denoted this method *Greedy* because it actually does not test if the Reynolds cavitation criteria is satisfied. For a coarse resolution the method might move x_{out} in too big steps – a failure that the method will not correct. However, if the inaccuracy of the result can be accepted the method is very efficient.

4.4.4 Newton iteration

One can interpret the condition $\mathrm{d}p/\mathrm{d}x = 0$ in the Reynolds cavitation criteria as a zero finding problem. Written with discrete variables we want to find a zero for

$$f = \frac{3p_N - 4p_{N-1} + p_{N-2}}{2\Delta x} \tag{4.32}$$

The zero finding can be carried out using the Newton-Raphson method. The gradient of f can be approximated by finite differences, or it may be calculated from a sensitivity analysis similar to the one presented in section 4.3. We have used the latter option which is more efficient. In figure 4.5 we have shown how

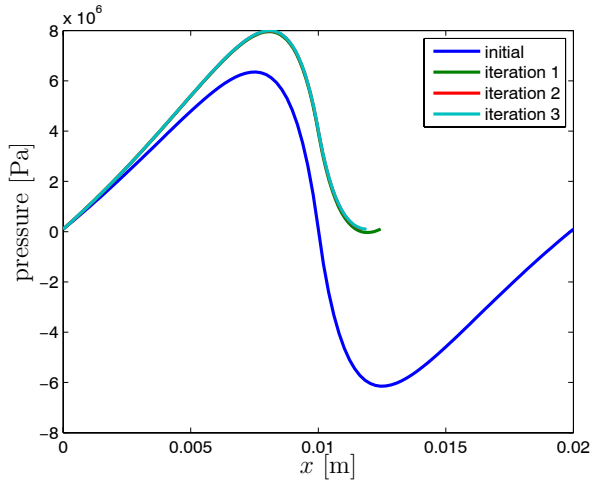

Figure 4.5: Cavitation method "Newton iteration". Only a few iterations are needed to find x_{outlet} accurately.

the iterations proceed. First we solve the Reynolds equation on the domain $x \in [0, L_{cd}]$. The point x_i where minimum pressure is obtained, is taken as the starting guess for the subsequent Newton iterations.

A great advantage of this method is that the Reynolds cavitation criteria is enforced directly. This means that upon convergence x_{out} is found accurately (remeshing of the interval $[0, x_{\text{out}}]$ is done in every iteration).

We did, however, also experience some problems with this method. In some cases the update becomes so small that the current x_{out} will remain unchanged due to the finite precision the of computer arithmetics. This can happen before the desired tolerance on $\mathrm{d}p/\mathrm{d}x$ is reached. It should be noted that this effect was encountered for rather pathological problems only (e.g. a *very* starved piston ring).

4.4.5 Bisection method

Finally we present the bisection method. This method works by successive divisions of an interval known to contain a x_{out}, which will satisfy $\mathrm{d}p/\mathrm{d}x = 0$. The initial search interval can be taken as the original interval $[0, L_{\text{cd}}]$, or any other better estimate. In each iteration either end point of the search interval is updated depending of the sign of $\mathrm{d}p/\mathrm{d}x$, so that the interval is halved in each iteration (more details are given in the next section). In figure 4.6 we have shown the first few iterations.

The bisection method enforces the Reynolds cavitation criteria explicitly,

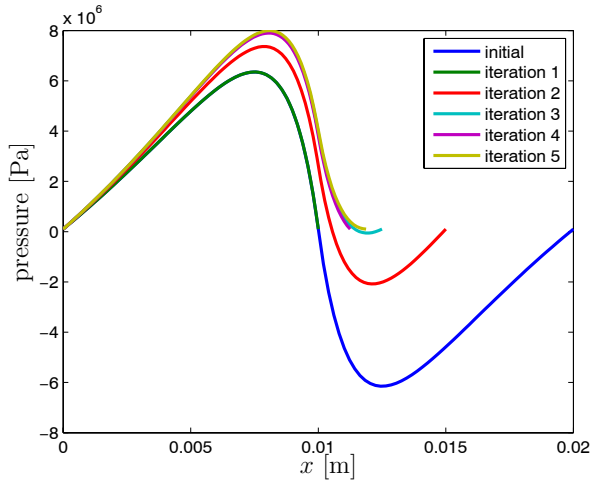

Figure 4.6: Cavitation method "bisection". Many iterations are needed.

which means that x_{out} will be located accurately. Another advantage is that the method is finite – it cannot iterate indefinitely (unless the stopping criteria is badly chosen).

The main disadvantage is that the method needs relatively many iterations, compared to the Greedy algorithm and Newton iterations.

4.5 Cavitation iteration – actual implementation

In the previous sections we gave a brief description of various ways of implementing the Reynolds cavitation criteria. Most of the methods can be combined in order to obtain a more accurate solution and/or improve stability.

In this thesis we apply the Reynolds equation for a few special cases only. Because of the restrictions mentioned earlier in sections 2.6 and 2.7 it is guarantied that at most one cavitation zone exists. Furthermore we are interested in the pressure distribution of the first subdomain with $p > p_{\text{cav}}$ only. This means that we can choose between methods "Greedy", "Newton", and 'Bisection" without disadvantage. In principle we would prefer the Newton method, because it has fast convergence and at the same time delivers an accurate solution. However, we will sometimes be solving badly scaled problems for which convergence of the Newton method is problematic. Therefore we have chosen to use the Greedy algorithm, and polish up the solution using the Bisection algorithm.

Thus we have implemented a cavitation algorithm in the following way

1. Calculate pressure distribution
2. If $p_{\min} = p_i > p_{\text{cav}}$ return $\mathbf{p}$, else
3. Set $x_{\text{out}} = x_i$ (where $p_i = p_{\min}$)
4. Calculate pressure distribution
5. Repeat from 3 until $x_i = x_N$, then
6. Set $x_{\max} = x_N$
7. Increment $x_{\max}$ with Δx
8. Calculate pressure distribution
9. If $\mathrm{d}p/\mathrm{d}x < 0$ repeat from 7, else
10. Set $x_{\min} = x_N$
11. Decrement $x_{\min}$ with Δx
12. Calculate pressure distribution
13. If $\mathrm{d}p/\mathrm{d}x > 0$ repeat from 11, else
14. Set $x_{\text{out}} = (x_{\min} + x_{\max})/2$
15. Calculate pressure distribution
16. If $\mathrm{d}p/\mathrm{d}x < 0$ set $x_{\min} = x_{\text{out}}$ else set $x_{\max} = x_{\text{out}}$
17. If $x_{\max} - x_{\min} < \varepsilon$ return $\mathbf{p}$
18. Repeat from 14

The bisection method is located in steps from 14 to 18. The stopping tolerance can be taken as $\varepsilon = 10^{-8}$. The pressure gradient is approximated by

$$\frac{\mathrm{d}p}{\mathrm{d}x} = \frac{3p_N - 4p_{N-1} + p_{N-2}}{2\Delta x} \tag{4.33}$$

It is important to use a second order approximation and not just a first order approximation (see section 5.3).

Chapter 5

Results from the Reynolds equation

In this chapter we present some solutions of the Reynolds equation applied to selected problems.

First we consider the fixed incline slider bearing. Three different types of solutions are compared 1) analytical, 2) numerical (incompressible), and 3) numerical (compressible). The convergence properties of the numerical methods are examined. The velocity distribution is also computed.

The next section contains an investigation of the prerequisites for the appearance of backflow in a lubricated conjunction. It will be seen that the point of bifurcation (backflow or no backflow) can be described analytically.

The final section is devoted to the relation between the minimum film thickness under a piston ring, the undisturbed film thickness on the liner, and the location of the inlet point on the piston ring. Only steady state conditions are considered. It will be seen that under certain circumstances it is possible to have multiple equilibrium points.

5.1 Fixed incline slider bearing

Consider a fixed incline slider bearing with the following properties

$$\begin{aligned} h_{\text{in}} &= 52 \;\; \text{inlet film thickness, } \mu\text{m} \\ h_{\text{out}} &= 12 \;\; \text{outlet film thickness, } \mu\text{m} \\ L &= 3 \;\; \text{slider width, mm} \\ U &= 5 \;\; \text{horizontal velocity of lower part, m/s} \\ \mu &= 0.05 \;\; \text{dynamic viscosity, Pa s} \\ p_{\text{in}} &= 101500 \;\; \text{pressure at inlet, Pa} \\ p_{\text{out}} &= 101500 \;\; \text{pressure at outlet, Pa} \\ \rho &\;\; \text{to be defined in the following sections, kg/m}^3 \end{aligned} \tag{5.1}$$

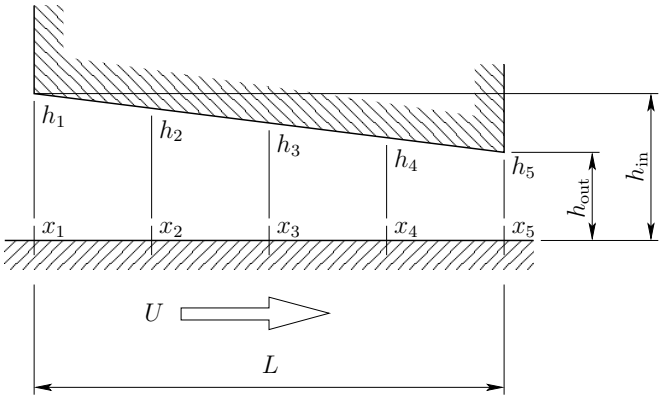

Figure 5.1: The fixed incline slider problem (5.1). Variables for discretization with 5 nodes is indicated. Graphics is not to scale.

5.1.1 Analytical, incompressible, and compressible solution

In figure 5.1 the problem is shown schematically. For the case of constant density and viscosity it is possible to solve Reynolds equation (4.1) analytically. Note that when density is constant, it can be eliminated from the Reynolds equation. The resulting pressure distribution is given analytically by [22]

$$p = \mu U L s_h \frac{6x(L-x)}{(Lh_{\text{out}} + Ls_h - s_h x)^2 (s_h + 2h_{\text{out}})} \tag{5.2}$$

where $s_h = h_{\text{in}} - h_{\text{out}}$. In figure 5.1 a discretization using five nodes is also shown. In figure 5.2 we plot the analytical solution together with incompressible numerical solutions for different numbers of nodes. It is seen that the numerical solution approaches the analytical solution as the number of discretization points is increased. In figure 5.3 we plot the maximum absolute error E against the resolution Δx. We have defined E in the following way

$$E = \max |p_i - p(x_i)| \quad i = 1, 2, \ldots, N \tag{5.3}$$

where p_i denotes the numerical solution at node i and $p(x_i)$ is the analytical solution at x_i. Node i is located at x_i. The spacial resolution Δx is given by (4.4), restated here for convenience

$$\Delta x = \frac{L}{N-1} \tag{5.4}$$

with N being the number of discretization points. The slope of the linear part

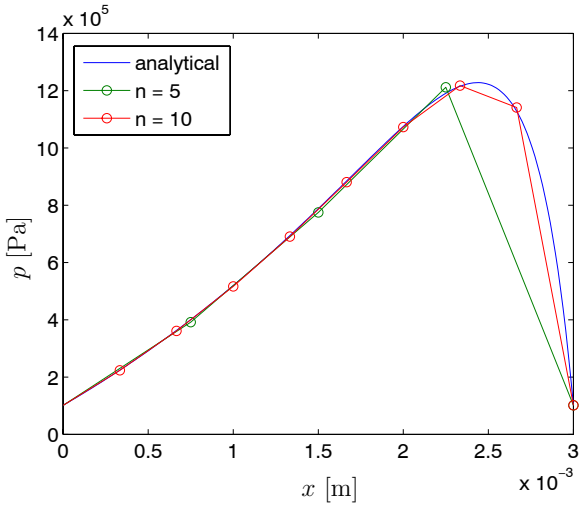

Figure 5.2: Analytical and numerical solutions of the fixed incline slider bearing (5.1). Data points 'o' are connected by line segments.

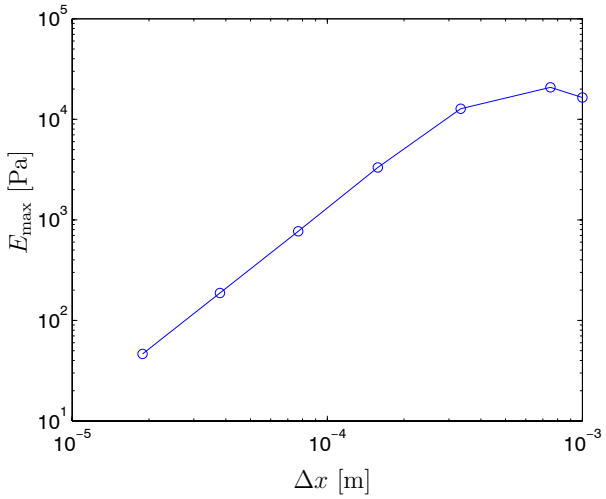

Figure 5.3: Maximum error between numerical and analytical solution for the fixed incline slider bearing problem (5.1). Data points 'o' are connected by line segments.

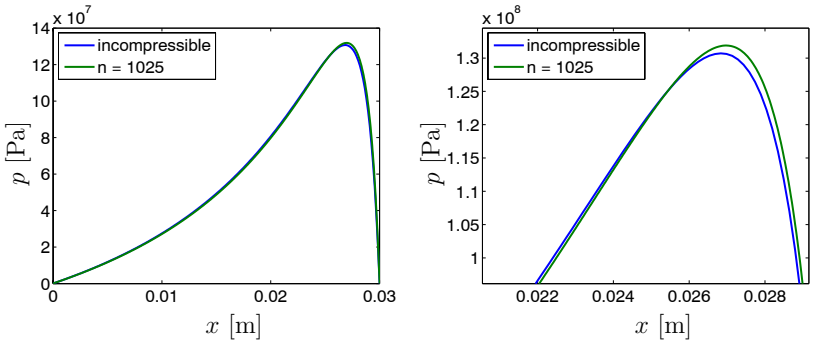

Figure 5.4: Left: Numerical solution ($N = 1025$) of the fixed incline slider bearing (5.5) with compressible fluid. The incompressible analytical solution is also shown for comparison. Right: A zoom on the pressure peak.

of the curve in figure 5.3 can be found by fitting a line to the data points. This has been done using Matlab's `polyfit` function [30], which gives a least squares fit. The slope is calculated to be $\alpha = 2.01$ which indicates that the discretization method (4.6)-(4.7) is of order 2. This result is in agreement with the second order approximations of spacial derivatives in (4.3).

We now solve the fixed incline slider problem using the compressible formulation. In order to see some effect of compressibility the parameters are changed in the following way

$$\begin{aligned} h_{\text{in}} &= 34 \quad \text{inlet film thickness, } \mu\text{m} \\ h_{\text{out}} &= 4 \quad \text{outlet film thickness, } \mu\text{m} \\ L &= 30 \quad \text{slider width, mm} \\ U &= 10 \quad \text{horizontal velocity of lower part, m/s} \\ \mu &= 0.05 \quad \text{dynamic viscosity, Pa s} \\ p_{\text{in}} &= 101500 \quad \text{pressure at inlet, Pa} \\ p_{\text{out}} &= 101500 \quad \text{pressure at outlet, Pa} \end{aligned} \tag{5.5}$$

Density is calculated using the equation of state (6.15).

In figure 5.4 we plot the numerical solution for $N = 1025$ along with the analytical solution of the incompressible case. It is seen that the pressure peak of the compressible solution is increased and shifted slightly downstream, as compared with the incompressible solution.

In figure 5.5 we plot the maximum error E against the spacial resolution Δx. Since we do not know the exact solution for the compressible case, we compute the error E by comparing with a high resolution numerical solution

$$E = \max |p_i^N - p_j^{1025}| \quad i = 1, 2, \ldots, N \quad (j = k | x_k = x_i) \tag{5.6}$$

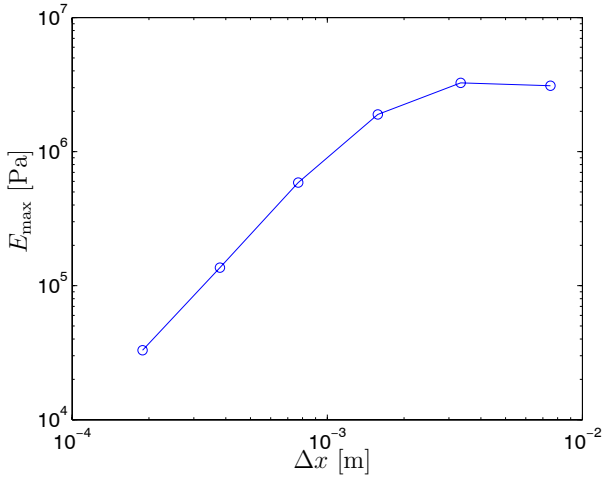

Figure 5.5: Maximum error between numerical solution and reference numerical solution ($N = 1025$) for the fixed incline slider bearing problem (5.1).

The numerical solution $\mathbf{p}^{1025}$ is obtained using $N = 1025$. Note that N is chosen so that the discretization points of $\mathbf{p}^N$ always coincide with a subset of the discretization points of the reference solution $\mathbf{p}^{1025}$.

Measuring the slope of the linear part in the same way as before gives $\alpha = 2.05$. This indicates that the discretization method (4.16)-(4.17) is of order 2.

Finally, we plot in figure 5.6 the number of iterations needed for different discretizations. It is seen that the number of iterations for this problem is almost constant regardless of the spacial resolution.

5.1.2 Velocity components

In this section we calculate the velocity components of the flow in the fixed incline slider bearing defined by (5.1). The first and second derivative of pressure with respect to x is needed, which we obtain from the analytical solution (5.2) by analytic differentiation

$$\begin{aligned} \frac{\mathrm{d}p}{\mathrm{d}x} &= 6\mu U L^2 s_h \frac{(s_h + h_{\text{out}})L - (s_h + 2h_{\text{out}})x}{(Lh_{\text{out}} + Ls_h - s_h x)^3 (s_h + 2h_{\text{out}})} \\ \frac{\mathrm{d}^2 p}{\mathrm{d}x^2} &= 12\mu U L^2 s_h \frac{(s_h^2 - h_{\text{out}}^2)L - s_h(s_h + 2h_{\text{out}})x}{(Lh_{\text{out}} + Ls_h - s_h x)^4 (s_h + 2h_{\text{out}})} \end{aligned} \tag{5.7}$$

Using the formulas (2.44) and (2.46) derived in section 2.10 we are able to calculate the velocity components. In figure 5.7 a surface plot of the u-velocity and

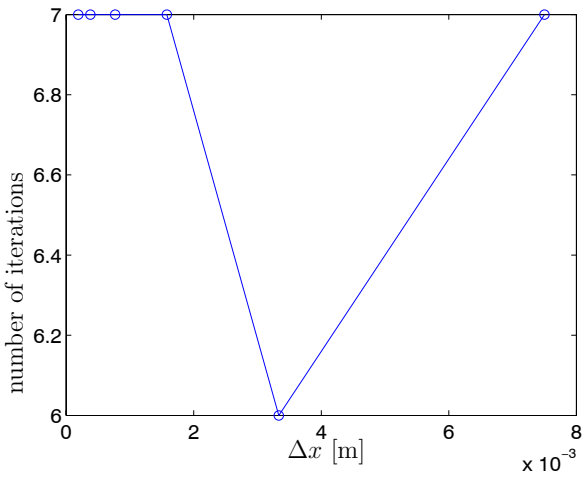

Figure 5.6: Number of iterations for the solution of (4.11) against the spacial resolution.

the v-velocity is shown. It is easily seen that backflow occurs for this bearing 1) the u-velocity is negative at the upper portion of the inlet, 2) the v-velocity is positive everywhere at the inlet. Note also how the fluid is accelerated at the outlet – the value of u becomes bigger than the sliding speed of the lower part.

5.2 Conditions for backflow

In this section we investigate the requirements for the presence of backflow in a lubricated conjunction. If we consider a piston ring, we might ask: Under what circumstances will all the oil on the liner pass under the piston ring? Or conversely stated: When will the piston ring be scraping oil, leading to build-up of oil in front of the piston ring? Our investigation is based on a system in the *steady state* condition. This means that the kinematics of the solids is fixed and that the oil film thickness is constant in the lubricated conjunction as well as outside the lubricated area. It will be seen that the results can be formulated in terms of the inlet height on the piston ring and the thickness of the undisturbed oil film on the liner. Furthermore the result turns out to be without direct dependence on the geometry of the piston ring and the pressure boundary conditions.

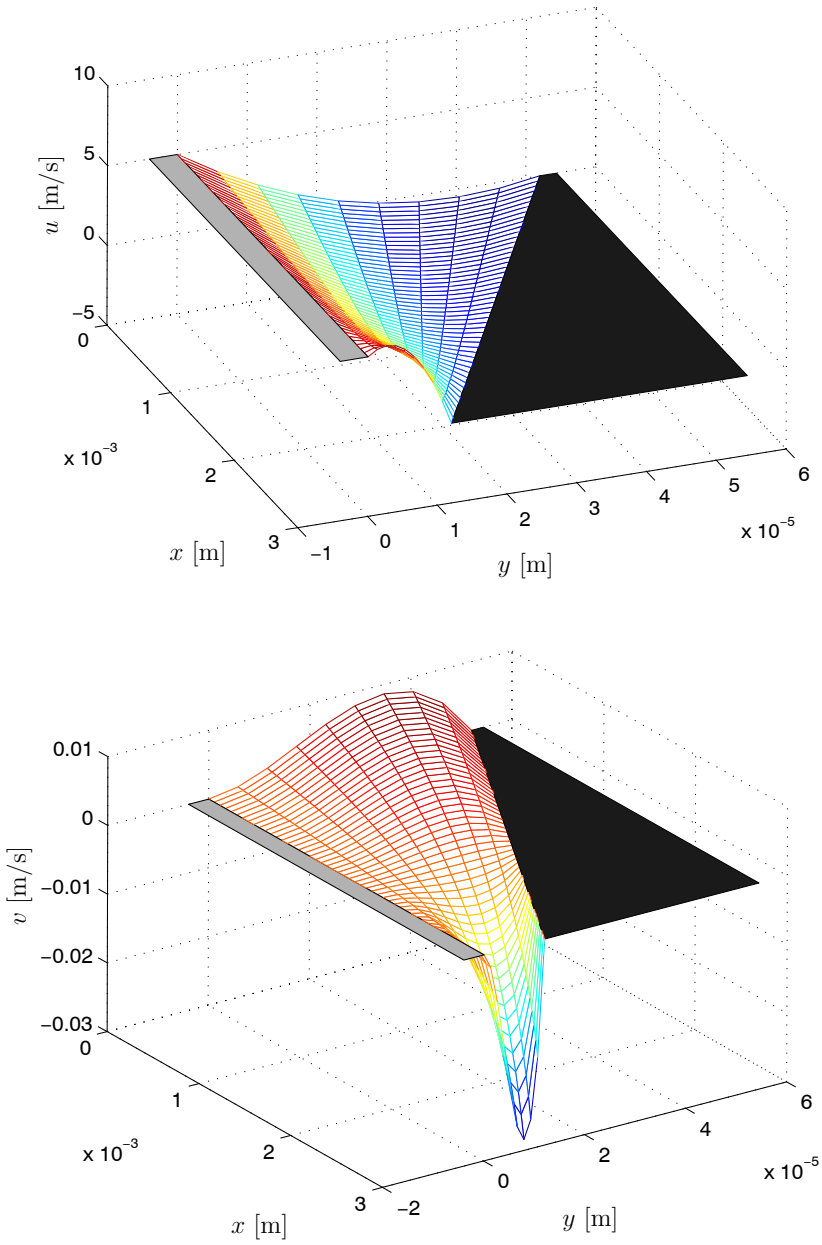

Figure 5.7: Velocity components for the fixed incline slider problem (5.1). The lightly shaded polygon indicates the lower (sliding) part of the bearing, while the heavy shaded polygon is the stationary upper part.

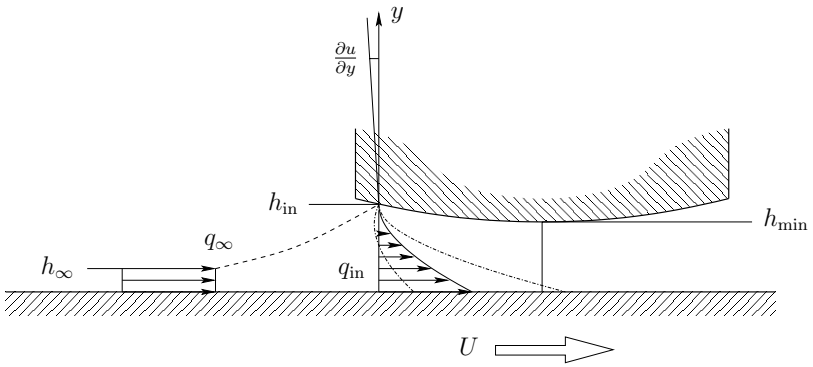

Figure 5.8: Definition of symbols, and indication of slope $\partial u/\partial y$.

We begin by defining some symbols to be used in the following (figure 5.8)

q_{in}	volume flow rate at inlet section
q_∞	volume flow rate at a section far away from the inlet section
u_{in}	x-component of velocity at inlet section
U	horizontal velocity of liner
h_{in}	distance between liner and piston ring at inlet
h_∞	thickness of undisturbed film on liner
h_{min}	minimum film thickness between liner and piston ring
$\mathrm{d}p/\mathrm{d}x$	gradient of pressure at inlet

We remind that when dealing with the Reynolds equation we always consider the upper part to be stationary, while the lower part may slide in the horizontal direction. This assumption has no physical impact in relation to the Reynolds equation.

In figure 5.8 we have also plotted different possible profiles of u at the inlet. The u-velocity at the inlet is given by (2.44), which in this context becomes

$$u_{\text{in}} = -y\left(\frac{h_{\text{in}} - y}{2\mu}\right)\frac{\mathrm{d}p}{\mathrm{d}x} + U\frac{h_{\text{in}} - y}{h_{\text{in}}} \tag{5.8}$$

Note that a zero exist at $y = h_{\text{in}}$ since the piston ring is stationary. The first step of our investigation is to determine when backflow occurs, i.e. under which conditions u_{in} becomes negative. It is seen that the velocity profile is a parabolic function in y. The derivative of u with respect to y becomes

$$\frac{\partial u}{\partial y} = \frac{2y - h_{\text{in}}}{2\mu}\frac{\mathrm{d}p}{\mathrm{d}x} - \frac{U}{h_{\text{in}}} \tag{5.9}$$

By continuity it is clear that the presence of negative u_{in} is related to the sign of $\partial u/\partial y$ at $y = h_{\text{in}}$, see figure 5.8. If the slope is negative we have no backflow, and if the slope is positive some level of backflow is present. Thus, the bifurcation point we are looking for is given by

$$\frac{\partial u}{\partial y}\Big|_{y=h_{\text{in}}} = \frac{h_{\text{in}}}{2\mu}\frac{\mathrm{d}p}{\mathrm{d}x} - \frac{U}{h_{\text{in}}} = 0 \tag{5.10}$$

Solving this equation for $\mathrm{d}p/\mathrm{d}x$ gives

$$\frac{\mathrm{d}p}{\mathrm{d}x} = \frac{2\mu}{h_{\text{in}}^2}U \tag{5.11}$$

We will use this result shortly, but first we consider the volume flow rate at the inlet. It is given by (2.42), which takes the form

$$q_{\text{in}} = -\frac{h_{\text{in}}^3}{12\mu}\frac{\mathrm{d}p}{\mathrm{d}x} + \frac{U}{2}h_{\text{in}} \tag{5.12}$$

We see that the flow is governed by a pressure term (Poiseuille) and a kinematic term (Couette). A positive pressure gradient will reduce the flow rate, while a negative pressure gradient will increase the flow rate.

Far away from the inlet section it is reasonable to assume that the fluid has a rigid body motion. Thus the velocity profile at a section is uniform and the volume flow rate becomes (figure 5.8)

$$q_\infty = h_\infty U \tag{5.13}$$

Now for a steady state condition to exist the volume flow rate at the inlet must equal the volume flow rate far away from the piston ring

$$q_{\text{in}} = q_\infty \tag{5.14}$$

Let us solve this equation for two special cases

1. $\mathrm{d}p/\mathrm{d}x$ given by (5.11)
2. $\mathrm{d}p/\mathrm{d}x = 0$

The first result 1) is found by combination of (5.11), (5.12), (5.13), and (5.14). After simplification we get

$$h_{\text{in}} = 3h_\infty \tag{5.15}$$

This equation shows that at the limit of backflow, the inlet film thickness will be three times bigger than the undisturbed film thickness. If the ratio $h_{\text{in}}/h_\infty < 3$ backflow cannot appear (under steady state conditions).

The second result 2) is found by substitution of $\mathrm{d}p/\mathrm{d}x = 0$ into (5.12) and further combination with (5.13) and (5.14). After manipulation the result is

$$h_{\text{in}} = 2h_{\infty} \tag{5.16}$$

This equation shows that in the absence of a pressure gradient only the Couette term will contribute to the transport of oil at the inlet. Clearly no backflow will appear since the velocity profile at the inlet becomes linear.

This concludes our investigation of the conditions under which backflow may appear in a lubricated conjunction. A single discriminating condition without direct dependence on geometry was found (i.e. no direct dependence on $\mathrm{d}h/\mathrm{d}x$ at $x = x_{\text{in}}$). If $h_{\text{in}}/h_{\infty} < 3$ it is not possible to have backflow under steady state conditions. On the other hand, if $h_{\text{in}}/h_{\infty} > 3$ and a steady state exists, then a zone of recirculation must exist in front of the piston ring.

The next obvious question is whether the bifurcation point is stable or not. If q_{in} decreases when the inlet point x_{in} is perturbed toward the center of the piston ring, then the inlet point is stable – otherwise it is unstable [29]. Therefore the stability of the inlet point is governed be the sign of $\partial q_{\text{in}}/\partial x_{\text{in}}$. We treat this question further in the next section.

We have sketched three different oil film configurations in figure 5.9

5.3 Relating h_{∞}, h_{in}, and h_{min}

In this section we investigate the properties of *starved* running conditions for piston rings. A starved running condition is characterized by the location of the inlet point on the piston ring. If the running surface of the piston ring is not fully immersed in oil the piston ring is said to suffer from starvation. This is opposed to the *fully flooded* running condition, where an excess of oil is available, see figure 5.10.

The location of the inlet point x_{in} on the piston ring is very important for the correct use of Reynolds equation. The aim of this section is to relate h_{∞}, h_{in}, and the minimum film thickness under the piston ring h_{min}. This will be done for a specific geometry corresponding to a certain piston ring described by

$$h = \frac{(x - C_r W_{\text{ring}})^2}{r_p} \tag{5.17}$$

where

$$\begin{aligned} x &= [0, W_{\text{ring}}] \\ r_p &= 0.5 \;\; \text{shape of ring, m} \\ C_r &= 0.5 \;\; \text{offset, [-]} \\ W_{\text{ring}} &= 12.5 \cdot 10^{-3} \;\; \text{width of piston ring, m} \end{aligned} \tag{5.18}$$

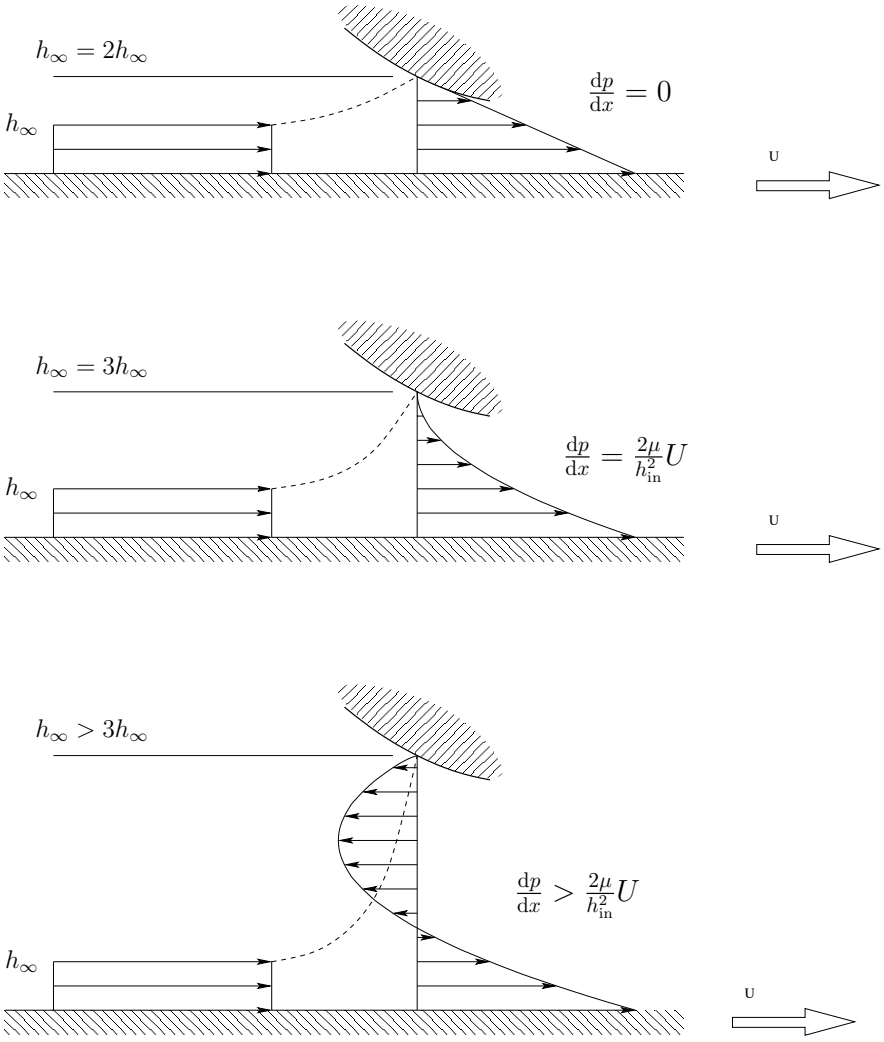

Figure 5.9: Results for back flow investigation

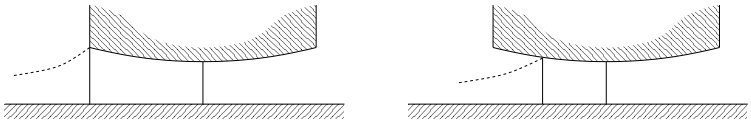

Figure 5.10: Left: Fully flooded. Right: Starved running conditions.

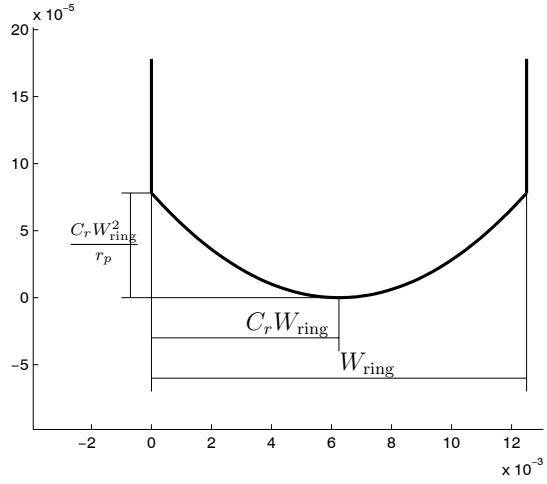

Figure 5.11: The shape of the piston ring given in (5.18). Aspect ratio not 1:1.

The shape of the piston ring is shown in figure 5.11 along with the definition of parameters describing geometry.

We begin by calculating q_{in} as a function of x_{in} with all other parameters held fixed, using the incompressible Reynolds equation. The volume flow rate is then given by (5.12).

$$q_{\text{in}} = -\frac{h_{\text{in}}^3}{12\mu}\frac{\mathrm{d}p}{\mathrm{d}x} + \frac{U}{2}h_{\text{in}}$$

The derivative $\mathrm{d}p/\mathrm{d}x$ entering this expression is approximated by finite differences. It should be noted that the difference scheme must be at least second order accurate. Below we quote three difference schemes of order 1, 2, and 3 respectively [21]

$$\begin{aligned}
\frac{\mathrm{d}p}{\mathrm{d}x} &= \frac{p_2 - p_1}{\Delta x} + O\left(\Delta x\right) \\
\frac{\mathrm{d}p}{\mathrm{d}x} &= \frac{-3p_1 + 4p_2 - p_3}{2\Delta x} + O\left((\Delta x)^2\right) \\
\frac{\mathrm{d}p}{\mathrm{d}x} &= \frac{-11p_1 + 18p_2 - 9p_3 + 2p_4}{6\Delta x} + O\left((\Delta x)^3\right)
\end{aligned} \tag{5.19}$$

The result of using either of these formulas is shown in figure 5.12 for the following input parameters

$$\begin{aligned}
p_{\text{in}} &= 101500 \text{ Pa} \\
p_{\text{out}} &= 101500 \text{ Pa} \\
p_{\text{cav}} &= 101500 \text{ Pa} \\
U &= 10 \text{ m/s} \\
\mu &= 0.05 \text{ Pa s} \\
h_{\text{min}} &= 10 \ \mu\text{m}
\end{aligned}$$

The volume flow rate q_{in} is calculated at 50 locations of the inlet point, given by

$$x_{\text{in}} = \frac{0.4W_{\text{ring}}}{49}(i-1) \quad i = 1, 2, 3, \ldots, 50 \tag{5.20}$$

We have calculated the flow rate using two discretizations of Reynolds equation, one with $N = 100$ and the other with $N = 1000$. From figure 5.12 it is seen that the first order approximation of $\mathrm{d}p/\mathrm{d}x$ converges too slowly to be of practical use. In fact the first order scheme does not even display the correct trend of the flow rate variation with respect to x_{in} for $N = 100$. Therefore we use the second order approximation in the following.

We now define (5.14) as a residual expression in the following way

$$R = q_{\text{in}} - q_\infty = -\frac{h_{\text{in}}^3}{12\mu}\frac{\mathrm{d}p}{\mathrm{d}x} + \frac{U}{2}h_{\text{in}} - Uh_\infty \tag{5.21}$$

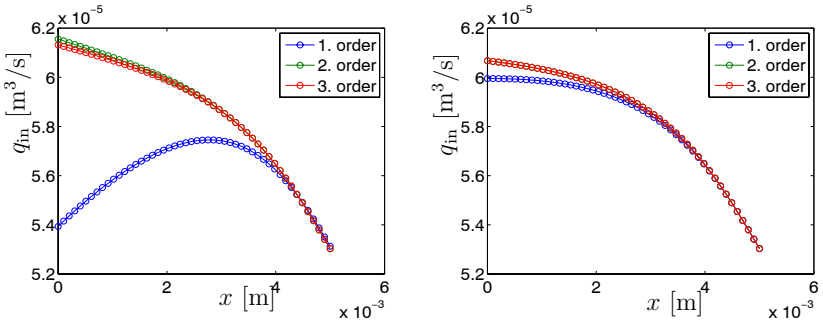

Figure 5.12: Volume flow rate q_{in} evaluated at 50 evenly distributed inlet points $x_{\text{in}} = 0, x_1, x_2, \ldots, 0.4W_{\text{ring}}$, using different orders for finite difference approximation of $\mathrm{d}p/\mathrm{d}x$. Left: discretization of Reynolds equation with 100 nodes. Right: discretization of Reynolds equation with 1000 nodes.

If for some combination of x_{in}, h_∞, h_{in}, h_{min}, p_{in}, p_{out}, p_{cav}, U, and μ it happens that $R = 0$, then we know that a *steady state* flow situation exists. We have investigated R for two different sets of parameters given by

Set 1

$$
\begin{aligned}
p_{\text{in}} &= 101500 \text{ Pa} \\
p_{\text{out}} &= 101500 \text{ Pa} \\
p_{\text{cav}} &= 101500 \text{ Pa} \\
U &= 10 \text{ m/s} \\
\mu &= 0.05 \text{ Pa s} \\
h_{\text{min}} &= 5 \ \mu\text{m} \\
h_\infty &= 3 \ \mu\text{m}
\end{aligned}
$$

Set 2

$$
\begin{aligned}
p_{\text{in}} &= 4060000 \text{ Pa} \\
p_{\text{out}} &= 101500 \text{ Pa} \\
p_{\text{cav}} &= 101500 \text{ Pa} \\
U &= 10 \text{ m/s} \\
\mu &= 0.01 \text{ Pa s} \\
h_{\text{min}} &= 15 \ \mu\text{m} \\
h_\infty &= 12.6 \ \mu\text{m}
\end{aligned}
$$

In both cases we let x_{in} vary as given by (5.20) and plot the value of R in figure 5.13. It is seen that for Set 1 $R = R(x_{\text{in}})$ has a single zero, while for Set 2 $R = R(x_{\text{in}})$ has two zeros. This shows that the number of solutions might not

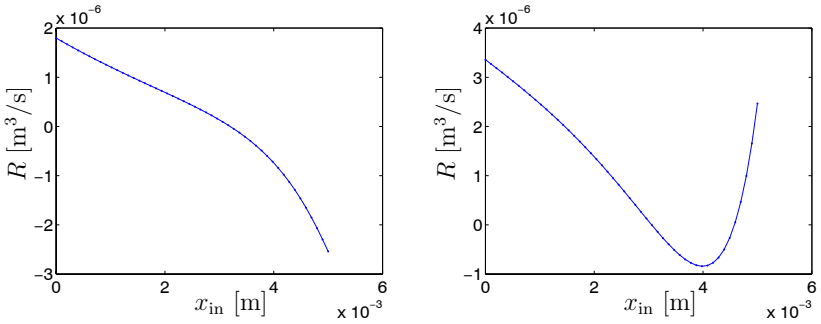

Figure 5.13: Residual (5.21). Left: only one zero exists. Right: two zeros exist.

be constant. At least in principle, the presence of multiple roots means that the system might show effects of hysteris. It is also seen that the slope of $\partial R/\partial x_{\text{in}}$ can be either negative or positive. When $\partial R/\partial x_{\text{in}}$ is negative the inlet point x_{in} is stable, otherwise it is unstable.

We now arrive at the main investigation of this section. The idea is to consider (5.21) as an equation $R = 0$. In this case we can select one parameter to be an implicit function of the remaining parameters. This is what we *really* want – to calculate x_{in} as a function of the remaining parameters

$$x_{\text{in}} = x_{\text{in}}(h_\infty, h_{\min}, p_{\text{in}}, p_{\text{out}}, p_{\text{cav}}, U, \mu, \rho) \tag{5.22}$$

Once x_{in} is found it is simple to calculate h_{in}.

If exactly one solution x_{in} exists, then we can find it by e.g. a bisection method. This is attractive because the method is very robust. However, since multiple solutions might exist bisection is not safe. Instead we use a gradient based method, namely Newton-Raphson iteration. This allows us to provide an initial guess for the location of the inlet point, and (at least in principle) follow a specific solution branch. In order to use the Newton-Raphson method we need the gradient of (5.21). Let us first define the discrete approximations of some derivatives

$$\begin{aligned} \frac{\mathrm{d}h_1}{\mathrm{d}x_1} &= \frac{-3h_1 + 4h_2 - h_3}{2\Delta x} \\ \frac{\mathrm{d}p_1}{\mathrm{d}x_1} &= \frac{-3p_1 + 4p_2 - p_3}{2\Delta x} \end{aligned} \tag{5.23}$$

Note that $x_{\text{in}} = x_1$, so in the following we will write x_1 instead of x_{in}. In terms of these expressions the discrete version of (5.21) becomes

$$R = -\frac{h_1^3}{12\mu}\frac{\mathrm{d}p_1}{\mathrm{d}x_1} + \frac{U}{2}h_1 - Uh_\infty \tag{5.24}$$

The gradient may then be obtained by

$$\frac{\mathrm{d}R}{\mathrm{d}x_1} = -\frac{h_1^2}{4\mu}\frac{\partial h_1}{\partial x_1}\frac{\mathrm{d}p_1}{\mathrm{d}x_1} - \frac{h_1^3}{12\mu}\frac{\partial}{\partial x_1}\left(\frac{\mathrm{d}p_1}{\mathrm{d}x_1}\right) + \frac{U}{2}\frac{\partial h_1}{\partial x_1} \tag{5.25}$$

The partial derivatives indicate derivatives that most be found using the sensitivity analysis presented in section 4.3. We may use the results found there to get

$$\begin{aligned} \frac{\partial h_1}{\partial x_1} &= \frac{\mathrm{d}h_1}{\mathrm{d}x_1} \\ \frac{\partial}{\partial x_1}\left(\frac{\mathrm{d}p_1}{\mathrm{d}x_1}\right) &= \frac{\left(-3\dfrac{\partial p_1}{\partial x_1} + 4\dfrac{\partial p_2}{\partial x_1} - \dfrac{\partial p_3}{\partial x_1}\right)\Delta x + (-3p_1 + 4p_2 - p_3)\dfrac{1}{N-1}}{2(\Delta x)^2} \end{aligned} \tag{5.26}$$

Here $\mathrm{d}h_1/\mathrm{d}x_1$ is given by the expression in (5.23) and the partial derivatives $\partial p_1/\partial x_1$, $\partial p_2/\partial x_1$, and $\partial p_3/\partial x_1$ are taken from the solution of equation (4.21). Having now obtained the gradient of R with respect to x_{inlet} we can employ the Newton-Raphson method for efficient solution of (5.21). It is noted that we only want solutions where x_{in} belongs to the interval $[0, x_{\max}]$. Here $x_{\max}$ is taken as e.g. $0.9 \cdot C_r W_{\text{ring}}$ or some other value close to the location of the minimum film thickness on the piston ring. This ensures that the domain for the Reynolds equation is always non-empty.

Sometimes a solution does not exist for $x_{\text{in}} \in [0, x_{\max}]$. Typically we are able to detect this by checking the value of x_{in}. If x_{in} becomes negative we stop the iteration, and return $x_{\text{in}} = 0$. The interpretation of this situation is that the piston ring is fully flooded. If on the other hand x_{in} becomes greater than $x_{\max}$ we return $x_{\text{in}} = x_{\max}$, which we understand as the piston ring being fully starved. Finally, a third situation can occur, in which the Newton-Raphson algorithm gets stuck in a local minimum. In figure 5.14 we have shown a situation where this will happen. The parameters have these values

Set 3

$$\begin{aligned} p_{\text{in}} &= 4060000 \text{ Pa} \\ p_{\text{out}} &= 101500 \text{ Pa} \\ p_{\text{cav}} &= 101500 \text{ Pa} \\ U &= 10 \text{ m/s} \\ \mu &= 0.01 \text{ Pa s} \\ h_{\min} &= 15\ \mu\text{m} \\ h_{\infty} &= 12.4\ \mu\text{m} \end{aligned}$$

We detect a local minimum by monitoring the number of Newton-iterations. If more than 20 iterations is taken it is highly unlikely that a zero exist in the local

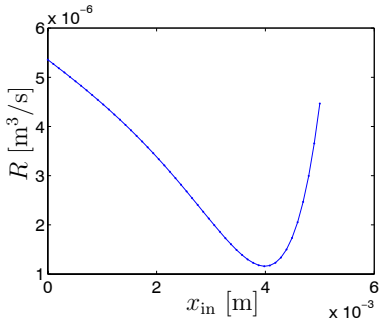

Figure 5.14: Example of a situation where the Newton-Raphson method gets caught in a local minimum.

neighborhood (it would have been found if it did exist). In this case we switch to a minimization algorithm in order to find the minimum value of R, that is, we try to do as good as we can in that situation. The minimization algorithm we use is known as the Golden Section Method, and has been implemented from the Numerical Recipes library [31].

We are now ready to calculate x_{in} for a concrete Diesel engine. The assumption we make is that the inlet height h_{in} can be determined in a quasi-static manner. At each location on the liner we feed the values of pressure, velocity, viscosity, minimum film thickness, and the undisturbed film thickness on the liner into (5.22). Then we solve for x_{in}, which tells us where the inlet point on the piston ring *would be for a steady state situation*. Thus, we assume that this "steady state" has much faster dynamics than the remaining part of the system. With these assumptions clarified we proceed with an example.

Piston kinematics are taken from the 4T50MX engine, as well as a pressure read-out from the combustion chamber and between the first (top) and second piston ring. These values are shown in figure 5.15 and are used as parameters for p_{in}, p_{out}, and U in (5.22). The remaining parameters are fixed at

$$\begin{aligned} h_{\min} &= 20 \ \ \mu\text{m} \\ h_\infty &= 12 \ \ \mu\text{m} \\ p_{\text{cav}} &= 101500 \ \ \text{Pa} \\ \mu &= 0.045 \ \ \text{Pa s} \\ \rho &= 900 \ \ \text{kg/m}^3 \end{aligned} \tag{5.27}$$

In figure 5.16 we have plotted h_{in} (calculated from the piston ring shape and the values of x_{in}) against the piston ring position on the liner. From the figure it is seen that the piston ring is fully flooded on the first 20 cm moving down from the top dead center. However, about at the same time as p_{in} has a peak the situation changes abruptly into starved conditions. The piston ring remains starved until

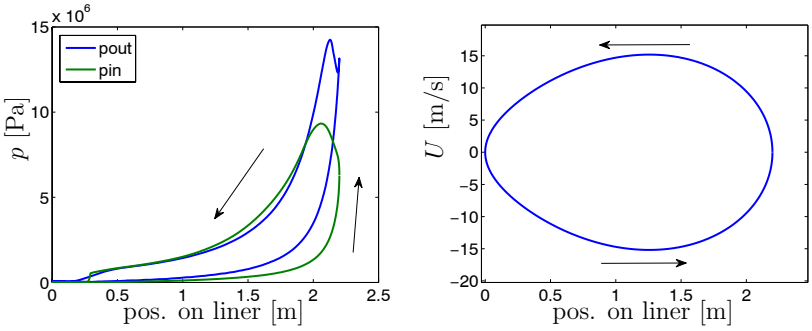

Figure 5.15: Left: Pressure at inlet and outlet as a function of piston ring location on liner. Right: Velocity of piston ring as a function of piston ring location on liner ($x = 2.2$ is top dead center).

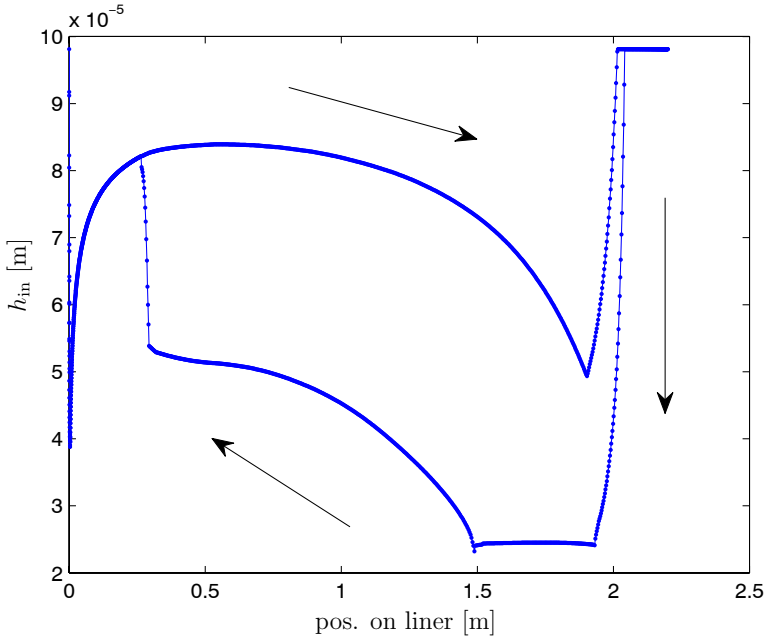

Figure 5.16: Inlet height h_{in} on piston ring as a function of piston ring location on liner (the top dead center is at $x = 2.2$ m).

a certain point after which the inlet height starts to increase. Again we observe a sudden change around 20 cm before the bottom dead center. It is possible that this behavior comes from the change in which of p_{in} or p_{out} is the greatest. On the other hand such a change in pressure did not occur at the top dead center, so it is suspected that the sliding speed of the piston is also very important. The return path from the bottom dead center to the top dead center has a more or less similar shape as when going down. The main difference is that the piston ring is less starved. This must be attributed to the smaller pressure levels during the compression cycle as compared with the combustion cycle.

Part II

Navier–Stokes equations

Chapter 6

The Navier–Stokes equations

This chapter is the beginning of the second part of this thesis. In this and the following chapters we will describe a simulation model for lubrication based on the Navier–Stokes equations. Before we dig into the governing equations, boundary conditions, discretization methods, etc we give an overview of the model. This will be helpful for the reading of the next chapters – one will know a little bit of what is to expect.

In the following we will often refer to the model by *the simulation program*. The simulation program is written in `C++` under Windows, using Microsoft Visual `C++` [32]. The program is designed to be a stand-alone program, without dependence on commercial software packages. The Numerical Recipes Library is used for common mathematical operations [31]. The sparse matrix package UMFPACK [33] is compiled under Windows using Cygwin [34]. The Automatically Tuned Linear Algebra Software [35], on which UMFPACK depends, is also compiled using Cygwin. A small graphics utility based on FreeGLUT [36] has been developed in order to allow graphical output directly from the simulation program. The program is easily run under Linux.

In figure 6.1 we have shown a sketch of the components of the model. As can be seen in the figure the simulation program supports various objects for the definition of a simulation problem. These objects are

solid Solids are used to represent rigid solid bodies. The geometry of a solid is defined with respect to a local coordinate system attached to the solid. A solid may be stationary or its kinematics can be prescribed by user-defined drivers. The number of solids in a simulation problem can be zero, one, or any other number as needed.

free surface Free surfaces are represented by a collection of nodes that are initially marked as belonging to the free surface. Any number of free surfaces can be defined, but merging or splitting of free surfaces during the simulation is not allowed. The evolution of a free surface is determined by the normal and shear forces on the free surface interface between the fluid do-

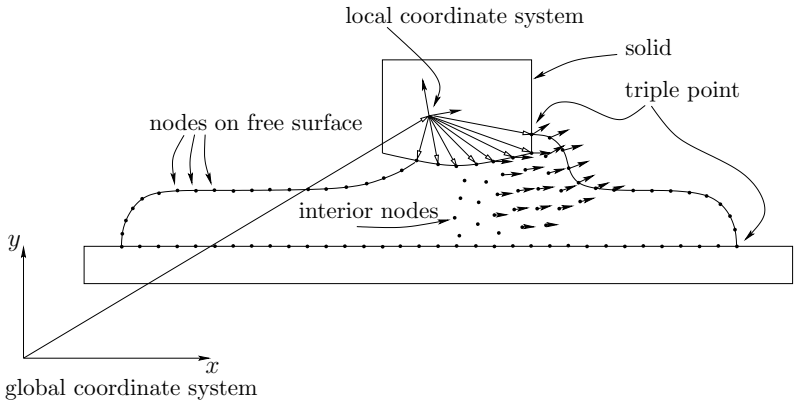

Figure 6.1: Sketch of the components of the simulation program.

main and void (the surroundings). A free surface can be closed (self-closed) or it may attach to solids.

fluid The simulation program allows the definition of a single fluid domain. The dynamics of fluid particles is governed by the Navier–Stokes equations. The boundary of the fluid domain may be composed of any number of solids and free surfaces.

triple point Triple points are used to implement the special boundary condition for the point where a solid, a free surface, and the surroundings touch. A triple point is considered as belonging to neither a solid nor a free surface. Instead a triple point will override the dynamics of the node on the solid and the free surface with which it coincides.

The objects mentioned above are the main building blocks available for the definition of a simulation problem. A number of settings are available, e.g. different settings for the equation of state, time integration method, etc. Hopefully this short presentation of the simulation program will make it easier to read the following chapters.

In the following we present the various equations used by the simulation program.

6.1 The Navier–Stokes equations

The Navier–Stokes equations express the rate of change of momentum for a fluid particle. Using tensor notation and Cartesian coordinates they may be stated as

[14]

$$\frac{\mathrm{d}v_i}{\mathrm{d}t} = \frac{1}{\rho}\frac{\partial \sigma_{ij}}{\partial x_j} \tag{6.1}$$

where $\mathrm{d}(\cdot)/\mathrm{d}t$ denotes the total derivative with respect to time (or substantial derivative), and

$$\sigma_{ij} = -p\delta_{ij} + \tau_{ij} \tag{6.2}$$

is the stress tensor. The symbol δ_{ij} is the Kronecker delta, and the shear stress τ_{ij} is given by

$$\tau_{ij} = \mu \varepsilon_{ij} \tag{6.3}$$

where μ is the dynamic viscosity, and ε_{ij} is the deformation rate tensor

$$\varepsilon_{ij} = \frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\delta_{ij} \tag{6.4}$$

For a 2D problem the combination of equations (6.1)-(6.4) gives the Navier–Stokes equations for a compressible fluid with constant viscosity

$$\begin{aligned} \frac{\mathrm{d}u}{\mathrm{d}t} &= \frac{1}{\rho}\left[-\frac{\partial p}{\partial x} + \mu\left(\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2}\right)\right] \\ \frac{\mathrm{d}v}{\mathrm{d}t} &= \frac{1}{\rho}\left[-\frac{\partial p}{\partial y} + \mu\left(\frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 v}{\partial x^2}\right)\right] \end{aligned} \tag{6.5}$$

In (6.1) and (6.5) we have written the left hand side of the momentum equations using the total derivative. This notation indicates that we are using Lagrangian coordinates, as mentioned in the introduction. Since, Lagrangian coordinates are associated with points that follow the deformation of the media, we have the following equations for position

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= u \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= v \end{aligned} \tag{6.6}$$

The continuity equation expresses the conservation of mass, and may be written in Lagrangian form as [14]

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \tag{6.7}$$

Finally an equation of state is needed to close the system. Any thermodynamic state can be uniquely described by two variables. For example, pressure can be

considered a function of specific volume (density) and temperature. In this work we do not consider the effects of temperature, which means that we can express pressure by a single variable, namely density

$$p = p(\rho)$$

In the next section we present the different equations of state that has been considered.

6.2 Equation of state

Since we are solving the Navier–Stokes equations in the compressible form an equation of state must be provided in order to close the system. Liquids have in general low compressibility, which is why is it customary to assume incompressibility for liquids (and in many cases also for gases, if the maximum flow speed is smaller than 1/3 of the speed of sound in the medium). However, the pressure levels in the oil film under a piston ring can become very high.

The question of compressibility or incompressibility has an important effect on the solution methods available for the Navier–Stokes equations. In the incompressible case the continuity equation (6.7) becomes a kinematic condition

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{6.8}$$

This situation makes the Navier–Stokes equations and the continuity equation (6.8) a *differential algebraic system*. It can be seen that the kinematic condition (6.8) corresponds to an infinitely high speed of sound in the medium.

The relationship between a general equation of state and the speed of sound a is given by [37]

$$a = \sqrt{\frac{\mathrm{d}p}{\mathrm{d}\rho}} \tag{6.9}$$

So if the speed of sound a is infinite, then it means that the derivative $\mathrm{d}p/\mathrm{d}\rho$ is also infinite. In this case it is impossible to express pressure as a function of density – no matter how we change pressure p there will be no change in density ρ, which is an obvious characteristic of an incompressible medium.

For a compressible medium we have a *finite* speed of sound. So in this case $\mathrm{d}p/\mathrm{d}\rho$ is never infinite, and a change in pressure implies a change in density (under isothermal conditions). The continuity equation given by (6.7) is a dynamic equation in contrast to the kinematic version (6.8). So the compressible Navier–Stokes equations are a set of differential equations, without a kinematic condition, which can be integrated in time directly. In this respect the compressible form is simpler than the incompressible form.

We have implemented four different equations of state 1) ideal gas law, 2) Monaghan power law, 3) Dowson-Higginson pressure-density relation, and 4) barotropic cavitation model. Each equation of state is described in the following.

6.2.1 Ideal gas law

The ideal gas law can be stated as

$$pV = nRT \tag{6.10}$$

or, under isothermal conditions

$$p = C_0 \rho \tag{6.11}$$

where

V volume, m^3
n number of molecules, mol
R universal gas constant, $8.31 \cdot 10^3$ kgmol/K
T temperature, K
C_0 appropriate gas constant, J/kg

We have implemented this expression mainly for testing of the simulation program. The special feature about the ideal gas law is that it gives a linear relation between pressure and density. Also, it is simple to change the speed of sound through the gas constant

$$a = \sqrt{C_0} \tag{6.12}$$

6.2.2 Monaghan power law

Monaghan has suggested the following expression for the simulation of liquids using the SPH method [38]

$$p = C_0 \left(\left[\frac{\rho}{\rho_0} \right]^{\gamma} - 1 \right) \tag{6.13}$$

where

C_0 constant related to speed of sound
ρ_0 reference density
γ constant

We have used this equation of state only in the initial phase of the development of the program for testing purposes.

6.2.3 Dowson-Higginson law

Based on physical measurements Dowson and Higginson suggest the following relationship between density and pressure for mineral oils [25]

$$\rho = \rho_0 \left(1 + \frac{0.6 p_{\text{gauge}}}{1 + 1.7 p_{\text{gauge}}} \right) \tag{6.14}$$

where

ρ_0 reference density at ambient pressure, kg/m^3
p_{gauge} gauge pressure, GPa

Rearranging for absolute pressure we get

$$p = \frac{\rho - \rho_0}{2.3\rho_0 - 1.7\rho} 10^9 + p_{\text{amb}} \tag{6.15}$$

where p_{amb} is the ambient pressure corresponding to the reference density ρ_0. This equation of state is the pressure-density relation we have used the most. Introducing the parameters

$$\begin{aligned} d_1 &= 2.3 \text{ [–]} \\ d_2 &= 1.7 \text{ [–]} \\ d_3 &= 10^9 \text{ Pa} \end{aligned} \tag{6.16}$$

allows us to put (6.15) in the following form

$$p = \frac{\rho - \rho_0}{d_1\rho_0 - d_2\rho} d_3 + p_{\text{amb}} \tag{6.17}$$

This expression is valid for $\rho \in [1; d_1/d_2) \cdot \rho_0$. Later on, we will need the two first derivatives of (6.17), but it seems natural to state the expressions here. We get

$$\frac{\mathrm{d}p}{\mathrm{d}\rho} = \frac{d_3(d_1 - d_2)\rho_0}{(d_1\rho_0 - d_2\rho)^2} \tag{6.18}$$

$$\frac{\mathrm{d}^2p}{\mathrm{d}\rho^2} = \frac{2d_2d_3(d_1 - d_2)\rho_0}{(d_1\rho_0 - d_2\rho)^3} \tag{6.19}$$

From this we can find the speed of sound using (6.9). We have plotted the pressure-density relation and the speed of sound, using parameters (6.16) in figure 6.2. From the figure it is seen that a modest change in density results in very large pressures. The speed of sound is in the range 1350 m/s to 2000 m/s for density in the range $\rho \in [900; 1000]$ kg/m^3.

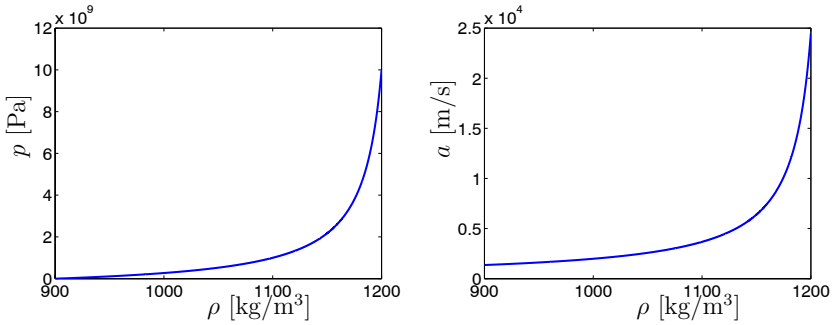

Figure 6.2: Left: The pressure-density relation due to Dowson and Higginson for mineral oil. Right: The corresponding speed of sound.

6.2.4 Cavitation model

It is well known that fluids in general cannot sustain tensile stresses. Instead a mechanism called *cavitation* takes place. This mechanism is a combination of two effects 1) expansion of dissolved microscopic bubbles of air and 2) instant evaporation of the pure fluid. The exact dynamics of cavitation is very complicated and form a field of research in its own. Two classes of modeling exist 1) multiphase models and 2) continuum models. With multiphase models the life of a bubble and the surrounding fluid is modeled directly as a two phase system. These models can be very sophisticated, at the expense of high demands on computing power. On the other hand the continuum models for cavitation treat bubbles and fluid as a single phase. These models are developed by making a number of assumptions about how a bubble grows and interacts with the fluid. A special class of these models is the *barotropic cavitation models*.

Barotropic means that pressure is assumed to depend on density only. These models are quite simple (compared to other models), but still useful for industrial applications. Because of the simplicity and almost vanishing cost of computing power, we choose a barotropic cavitation model. Wallis has developed the following expression for the speed of sound in a liquid-vapor mixture [39]

$$a = \sqrt{\frac{1}{(\alpha\rho_g + (1-\alpha)\rho_l)\left(\frac{\alpha}{\rho_g a_g^2} + \frac{1-\alpha}{\rho_l a_l^2}\right)}} \tag{6.20}$$

where

$$\begin{aligned} a &\quad \text{speed of sound, m/s} \\ \alpha = \frac{\rho - \rho_l}{\rho_g - \rho_l} &\quad \text{void fraction} \\ \rho_g &\quad \text{density of pure gas at ambient pressure, kg/m}^3 \\ \rho_l &\quad \text{density of pure liquid at ambient pressure, kg/m}^3 \\ a_g &\quad \text{speed of sound of pure gas at ambient pressure, m/s} \\ a_l &\quad \text{speed of sound of pure liquid at ambient pressure, m/s} \end{aligned} \tag{6.21}$$

Comparing with (6.9) the following differential equation can be obtained

$$\frac{\mathrm{d}p}{\mathrm{d}\rho} = \frac{1}{(\alpha\rho_g + (1-\alpha)\rho_l)\left(\dfrac{\alpha}{\rho_g a_g^2} + \dfrac{1-\alpha}{\rho_l a_l^2}\right)} \tag{6.22}$$

This equation can by integrated analytically from $\pi = p_{\text{amb}}$ to $\pi = p$ to give the following expression for pressure as a function of density

$$p = p_{\text{amb}} + p_{gl}\log\left[\frac{\rho_g a_g^2\left(\rho_l + \alpha(\rho_g - \rho_l)\right)}{\rho_l\left(\rho_g a_g^2 - \alpha(\rho_g a_g^2 - \rho_l a_l^2)\right)}\right] \tag{6.23}$$

where

$$p_{gl} = \frac{\rho_g a_g^2 \rho_l a_l^2 (\rho_g - \rho_l)}{\rho_g^2 a_g^2 - \rho_l^2 a_l^2}$$

Schmidt has combined this expression with the assumption of incompressibility once $\rho < \rho_g$ or $\rho > \rho_l$ [40]. In another paper Schmidt assumes constant speed of sound when $\rho > \rho_l$ [41].

Following these lines of thinking we propose a cavitation model where the ideal gas law is used when $\rho < \rho_g$, the expression based of Wallis' model for a liquid-gas mixture (6.21) is used when $\rho_g < \rho < \rho_l$, and the Dowson-Higginson model is used when $\rho > \rho_l$. While the three expressions can be pieced together to form a continuous pressure-density relation, the transition points lack differentiability. Therefore we insert a fourth order spline at the transition from pure gas to the liquid-gas mixture, and from the liquid-gas mixture to the pure liquid. Thus, the compound expression looks like this

$$p = \begin{cases} \text{Ideal Gas} & , \quad 0 < \rho \leq \rho_1 \\ \text{Spline} & , \quad \rho_1 < \rho \leq \rho_2 \\ \text{Wallis} & , \quad \rho_2 < \rho \leq \rho_3 \\ \text{Spline} & , \quad \rho_3 < \rho \leq \rho_4 \\ \text{Dowson-Higginson} & , \quad \rho_4 < \rho \end{cases} \tag{6.24}$$

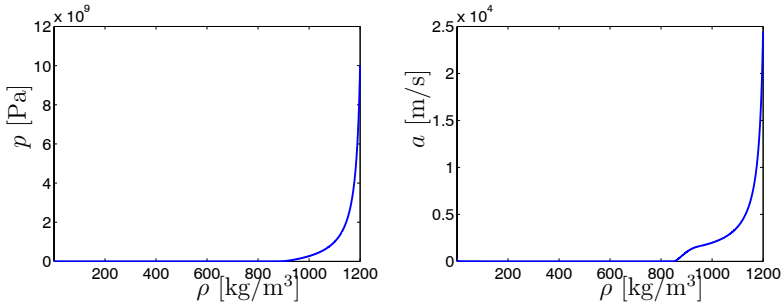

Figure 6.3: Left: The compound equation of state given in (6.24) for the modeling of cavitation. Right: The corresponding speed of sound.

where $\rho_1 < \rho_g < \rho_2$ and $\rho_3 < \rho_l < \rho_4$. The two splines are determined by requiring that $\mathrm{d}p/\mathrm{d}\rho$ and $\mathrm{d}^2p/\mathrm{d}\rho^2$ are continuous at points ρ_g^1 and ρ_g^2, and ρ_l^1 and ρ_l^2, respectively. In figure 6.3 we have plotted the resulting pressure-density relation and the speed of sound.

The viscosity of a liquid-gas mixture can be modeled by a linear interpolation between the viscosity of the pure liquid and the pure gas [42]

$$\mu = \alpha\mu_g + (1-\alpha)\mu_l \tag{6.25}$$

where

$$\begin{aligned} \alpha &\quad \text{void fraction as given in (6.21)} \\ \mu_g &\quad \text{viscosity of pure gas at ambient pressure, Pa s} \\ \mu_l &\quad \text{viscosity of pure liquid at ambient pressure, Pa s} \end{aligned} \tag{6.26}$$

However, in this work we have only considered constant viscosity as already indicated by the form of the Navier–Stokes equations (6.5). Therefore we have not implemented the interpolation method above.

Chapter 7

Multibody kinematics

In this chapter we describe some kinematic transformations related to multibody systems. We only consider rigid bodies in 2D. The expressions derived in this chapter are needed for the evaluation of positions, velocity components and accelerations of points on the boundary of a solid body.

In figure 7.1 we sketch a generic body embedded in the global coordinate system. In 2D the configuration of a rigid body is completely determined by six parameters, three values for position/orientation and three values for velocities/angular velocity. These coordinates are known as *body coordinates* and are associated with a local coordinate system rigidly attached to the body. Typically, one chooses origo of the local coordinate system to coincide with the body center of mass. This is, however, not a requirement.

It is convenient to express the shape of a body using the local body coordinate system. Because the body is rigid we can establish kinematic transformations between the local coordinate system and the global coordinate system. Referring to figure 7.1 we see directly that the following vector equation holds

$$\vec{R} = \vec{X} + \vec{r} \tag{7.1}$$

where $\vec{R}$ is the geometrical vector from O to a point of interest P on the body, $\vec{X}$ is the geometrical vector from O to o, and $\vec{r}$ is the geometrical vector from o to P. This vector equation can be transformed into an algebraic expression by referring all vectors to the global coordinate system. As mentioned previously it is convenient to express $\vec{r}$ in the local coordinate system, which means that a transformation from local coordinates to global coordinates is necessary.

This transformation is obtained by expressing the local basis vectors in terms of the global basis vectors. Given the angle of rotation ϕ, measured in the counterclockwise direction from the x-axis, this becomes the direction cosines

$$\begin{pmatrix} r_x \\ r_y \end{pmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{pmatrix} r_\xi \\ r_\eta \end{pmatrix} \tag{7.2}$$

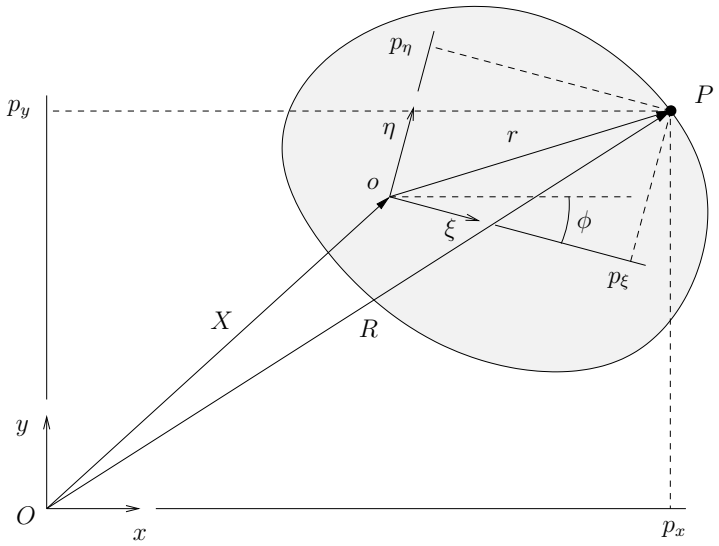

Figure 7.1: Definition of vectors for a multibody system.

where

r_ξ ξ-component of $\vec{r}$ (local coordinates)
r_η η-component of $\vec{r}$ (local coordinates)
r_x x-component of $\vec{r}$ (global coordinates)
r_y y-component of $\vec{r}$ (global coordinates)

With the expression above in mind, we may write

$$\begin{pmatrix} P_x \\ P_y \end{pmatrix} = \begin{pmatrix} x + r_\xi \cos\phi - r_\eta \sin\phi \\ y + r_\xi \sin\phi + r_\eta \cos\phi \end{pmatrix} \tag{7.3}$$

where

x x-component of o (global coordinates)
y y-component of o (global coordinates)
P_x x-component of P (global coordinates)
P_y y-component of P (global coordinates)

This expression gives the global coordinates for P given the location of the body (x, y, ϕ) and the local coordinates of r denoted by (r_ξ, r_η).

The velocity components at P are found by taking the time derivative of (7.3). We get

$$\begin{pmatrix} P_u \\ P_v \end{pmatrix} = \begin{pmatrix} u + (-r_\xi \sin\phi - r_\eta \cos\phi)\omega \\ v + (r_\xi \cos\phi - r_\eta \sin\phi)\omega \end{pmatrix} \tag{7.4}$$

where $\omega = \mathrm{d}\phi/\mathrm{d}t$. It is noted that the time derivative of r_x and r_y are zero, since the body is rigid. Finally, the acceleration is obtained by another differentiation of (7.4)

$$\begin{pmatrix} P_{ax} \\ P_{ay} \end{pmatrix} = \begin{pmatrix} a_x + (-r_\xi \cos\phi + r_\eta \sin\phi)\omega^2 + (-r_\xi \sin\phi - r_\eta \cos\phi)a_\phi \\ a_y + (-r_\xi \sin\phi + r_\eta \cos\phi)\omega^2 + (r_\xi \cos\phi - r_\eta \sin\phi)a_\phi \end{pmatrix} \quad (7.5)$$

The expressions above can be simplified by introducing (7.2). Summarizing we get

$$\begin{aligned} \begin{pmatrix} P_x \\ P_y \end{pmatrix} &= \begin{pmatrix} x + r_x \\ y + r_y \end{pmatrix} \\ \begin{pmatrix} P_u \\ P_v \end{pmatrix} &= \begin{pmatrix} u - r_y\omega \\ v + r_x\omega \end{pmatrix} \\ \begin{pmatrix} P_{ax} \\ P_{ay} \end{pmatrix} &= \begin{pmatrix} a_x - r_x\omega^2 - r_y a_\phi \\ a_y - r_y\omega^2 + r_x a_\phi \end{pmatrix} \end{aligned} \quad (7.6)$$

where

P_x	global x-coordinate of P
P_y	global y-coordinate of P
P_u	global x-component of velocity at P
P_v	global y-component of velocity at P
P_{ax}	global x-component of acceleration at P
P_{ay}	global y-component of acceleration at P
x	global x-position of body
y	global y-position of body
ϕ	global orientation of body
u	global x-component of velocity of body
v	global y-component of velocity of body
ω	global angular velocity of body
a_x	global x-component of acceleration of body
a_y	global y-component of acceleration of body
a_ϕ	global angular acceleration of body
r_x	x-component of r
r_y	y-component of r

More information on multibody kinematics and multibody dynamics may be found in monographs [19] and [20].

Chapter 8

Boundary conditions for the Navier–Stokes equations

In this chapter we describe the various boundary conditions that are used for the Navier–Stokes equations. As mentioned in the beginning of chapter 6 we have the following objects available for the boundary of the fluid domain 1) solids, 2) free surfaces, and 3) triple points. In the following each type of boundary is described in its own section. One exception is made, since we have treated the evaluation of pressure on the boundary in a section by itself.

8.1 Solid boundaries

On solid bodies we use the no-slip condition. The physical meaning of this is that there can be no relative velocity between particles on the solid and fluid particles touching the solid. Therefore, the velocity field of the fluid must equal the velocities on the solids at the position of the solid boundaries. This is expressed by

$$\begin{aligned} u_{\text{fluid}} &= u_{\text{wall}} \\ v_{\text{fluid}} &= v_{\text{wall}} \end{aligned} \tag{8.1}$$

valid at the positions of the solid boundaries. It is assumed that the solids can be modeled as rigid bodies. Section 7 describes the kinematic transformations needed to obtain velocities anywhere on a rigid body given the kinematics of a fixed coordinate system attached to the body.

In order to evaluate forces on solids the pressure on the solid boundary is needed. A boundary condition for pressure is the subject of the next section.

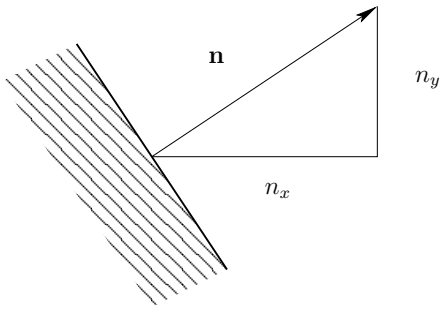

Figure 8.1: A sketch of the boundary and a unit normal vector.

8.2 Pressure boundary condition

The development of boundary conditions for pressure for the Navier–Stokes equations can be considered a field in its own. In this section we discuss boundary conditions for pressure on the parts of the boundary where pressure is not explicitly known. For example, at an inflow the pressure should be prescribed from some value of interest. On the other hand, pressure is not known a priori on solid boundaries.

For the incompressible Navier–Stokes equations various boundary conditions have been proposed. They show up in the Poisson equation usually employed to calculate the corrections of the velocity field, in order to satisfy the kinematic condition of a divergence free velocity field. In a way these boundary conditions are developed with certain physical as well as mathematical properties in mind. One simple model is to assume a Neumann condition for pressure, given by $\partial p/\partial \mathbf{n} = 0$. This notations means that $n_x \partial p/\partial x + n_y \partial p/\partial y = 0$, where $\mathbf{n}$ is a unit vector normal to the boundary (figure 8.1). However, more advanced boundary conditions exist [27]. For the compressible Navier–Stokes equations the situation is a little different, since a boundary condition for pressure is actually not needed. However, since we would like to know the pressure at e.g. a solid boundary, we need to establish a boundary condition anyway. We have considered two methods 1) extrapolation from interior points, and 2) Neumann condition derived from momentum equations.

8.2.1 Extrapolation

For compressible flows the standard procedure is to extrapolate pressure from the interior of the fluid domain. Thus based on values in the interior one can extrapolate values onto the boundary. Usually one would use linear (first order) or parabolic (second order) extrapolation.

The extrapolation process is based on fitting some functional expression to

known data points, and then evaluate this expression in the point where the value is needed. We will not provide the details here, because the extrapolation method depends on the layout of the known data points. This is related to the discretization method used for the governing equations, which is described in section 9.2.

8.2.2 Neumann condition

In this section we discuss another boundary condition for pressure, which is based on the momentum equations (6.5). The first observation is that the momentum equations should be satisfied everywhere in the computational domain – also on the boundary of the domain. The second observation is that the velocity components on the boundary should be known at all times – either prescribed explicitly or given implicitly by relevant equations. Therefore, on the boundary, (6.5) provide two equations in a single unknown which is the pressure p. Of course we cannot (in general) satisfy both equations at the same time, but we can formulate the following relaxed equation

$$\begin{aligned} &n_x \frac{\mathrm{d}u}{\mathrm{d}t} + n_y \frac{\mathrm{d}v}{\mathrm{d}t} = \\ &\quad n_x \frac{1}{\rho}\left[-\frac{\partial p}{\partial x} + \mu\left(\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2}\right)\right] \\ &+ n_y \frac{1}{\rho}\left[-\frac{\partial p}{\partial y} + \mu\left(\frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 v}{\partial x^2}\right)\right] \end{aligned} \tag{8.2}$$

This equation is obtained by taking the dot product of the momentum equations (6.5) and the boundary normal $\mathbf{n}$. We can rearrange (8.2) in the following way

$$\begin{aligned} \frac{\partial p}{\partial \mathbf{n}} &= -\rho\left(n_x \frac{\mathrm{d}u}{\mathrm{d}t} + n_y \frac{\mathrm{d}v}{\mathrm{d}t}\right) \\ &\quad + n_x\left[\mu\left(\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2}\right)\right] \\ &\quad + n_y\left[\mu\left(\frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 v}{\partial x^2}\right)\right] \end{aligned} \tag{8.3}$$

This has the form of a Neumann condition for p. It is noted that (8.2) and (8.3) are mathematically equivalent. In fact the term $\partial p/\partial \mathbf{n}$ must be expanded into $n_x \partial p/\partial x + n_y \partial p/\partial y$ before further processing of (8.3).

Looking at (8.2) we see that p as well ρ enters the equation. However, p and ρ are related through the equation of state. This means that we can either eliminate p or ρ using either the relation

$$\rho = \rho(p) \tag{8.4}$$

or

$$\begin{aligned} \frac{\partial p}{\partial x} &= \frac{\mathrm{d}p}{\mathrm{d}\rho}\frac{\partial \rho}{\partial x} \\ \frac{\partial p}{\partial y} &= \frac{\mathrm{d}p}{\mathrm{d}\rho}\frac{\partial \rho}{\partial y} \end{aligned} \tag{8.5}$$

We have decided to use the latter option, since we will be computing derivatives of density also at the interior of the simulation domain.

8.2.3 Comparison of methods

Before we end the section on boundary conditions for pressure we discuss some of the features of the two formulations. Beginning with the extrapolation method we have

- extrapolation is relatively easy to implement
- no iteration is needed
- only the density field is referenced – no coupling with the velocity field
- not based on modeling of physics

The Neumann condition on the other hand has the following properties

- more complicated to implement than the extrapolation method
- iteration is needed if the equation of state is nonlinear
- coupling between the density field and velocity field exist
- based on modeling of physics

From this overview we see that the two methods are complementary in all respects included in the list. From the modeling point of view we prefer the Neumann condition. However, with respect to implementation the extrapolation method is more attractive.

8.3 Free surface

The boundary condition for a free surface is that the normal stress and shear stress on either side of the interface must balance.

The stress vector at the interface is composed of different terms depending on the kind of interface. One part comes from the stress tensors associated with the media on each side of the interface. In the case of free surface modeling an additional contribution comes from the surface tension.

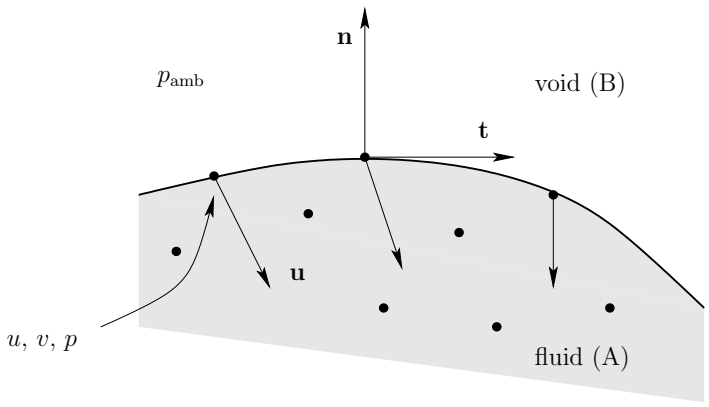

Figure 8.2: A free surface.

The physical origin of surface tension is the forces acting between molecules. At the interface between to different media a molecule will experience stronger attraction to either of the two media than the other. In continuum mechanics this effect is modeled by a force, which is at all times tangential to the interface. If the interface is a straight line (or a plane in 3D) there is thus no net effect from surface tension. If the curvature on the other hand is non-zero the result is a jump in normal stress at the interface. A free surface is sketched in figure 8.2. Surface tension can be described mathematically by

$$f = \kappa\sigma = \frac{x'y'' - y'x''}{\left((x')^2 + (y')^2\right)^{3/2}}\sigma \tag{8.6}$$

where

- κ is the curvature (normalized for unit speed parametrization), [-]
- x' first derivative of x with respect to curve length parameter τ, [-]
- x'' second derivative of x with respect to curve length parameter τ, [-]
- y' first derivative of y with respect to curve length parameter τ, [-]
- y'' second derivative of y with respect to curve length parameter τ, [-]
- τ curve length measured along free surface, m
- σ surface tension, N/m

If we combine the stress tensor part and the surface tension part, we get the following boundary conditions at the free surface interface

$$\begin{aligned} \sigma_{ij}^{\mathrm{A}} n_i n_j + \kappa\sigma &= \sigma_{ij}^{\mathrm{B}} n_i n_j \\ \sigma_{ij}^{\mathrm{A}} n_i t_j &= \sigma_{ij}^{\mathrm{B}} n_i t_j \end{aligned} \tag{8.7}$$

where

σ_{ij}	is the stress tensor
n_i	is the normal vector
t_i	is the tangent vector
A,B	denote media A and media B, respectively

If we for medium A consider a compressible fluid the stress tensor becomes (from (6.2))

$$\sigma_{ij}^{\mathrm{A}} = -p\delta_{ij} + \mu\left(\frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\delta_{ij}\right)$$

In this thesis it is assumed that the viscosity of air is negligible. So the stress tensor associated with the surrounding air (medium B) reduces to the following

$$\sigma_{ij}^{\mathrm{B}} = -p_{\mathrm{amb}}\delta_{ij}$$

The subscript $_{\mathrm{amb}}$ refers to ambient conditions. For 2D problems we have the following relationship between the normal vector and the tangent vector

$$n_x = -t_y \quad \text{and} \quad n_y = t_x$$

Substituting these expressions in (8.7) we get the following boundary conditions for the free surface interface

$$\begin{aligned} -p - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) + 2\mu\left(\frac{\partial u}{\partial x}n_x^2 + \frac{\partial v}{\partial y}n_y^2 + \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)n_x n_y\right) + \kappa\sigma &= -p_{\mathrm{amb}} \\ 2\left(\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right)n_x n_y + \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\left(n_y^2 - n_x^2\right) &= 0 \end{aligned} \tag{8.8}$$

where all terms are evaluated at the free surface. It is seen that we have two equations and three unknowns, namely u, v, and p. The pressure p on the left hand side of the first equation must somehow be established. In order to find the value of p we use either of the two methods given in the previous section. That is 1) extrapolation from the interior of the fluid domain, or 2) the Neumann condition. If we use method 1) the pressure is known explicitly, meaning that p is no longer an unknown in equation (8.8). In this case we then have a linear system of two equations for two unknowns. If we instead use method 2) we get a system of three equations with three unknowns. Depending on the equation of state this system of equations is linear or nonlinear.

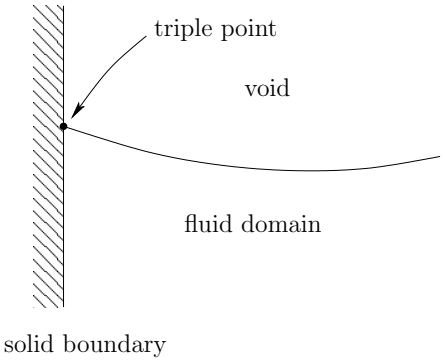

Figure 8.3: A triple point.

8.4 Triple point

The point where three different phases or media intersect is called a triple point. For our application triple points exist where a solid, a fluid, and the void region intersect, i.e. the attachment point of a free surface on a solid body, see figure 8.3

Triple points require special treatment because of what is known as the *kinematic paradox*. We have previously stated in section 8.1 that we use the no-slip condition on solid boundaries. On the other hand it is obvious that a triple point must be able to slide along the boundary of a solid. For example, when filling a glass with water the free surface as well as the contact line on the glass is moving, as water is poured in.

The physics for triple point action takes place at the level of molecules. The effect of attraction between molecules can be compared with the mechanism on the free surface, but the situation is more complicated because three media are present. The so called *wetting problem* is a field of research in its own involving experimental work as well as theoretical work. Still there is no fully justified large scale description available, but simplified models can be used.

In our application we assume that the solid boundaries are impermeable, which means that there can be no flow through a solid. In figure 8.4 we have sketched a moving solid boundary and a free surface. The requirement of no flow through the solid boundary can be written as

$$\mathbf{n}^{\text{solid}} \cdot \mathbf{u}^{\text{solid}} = \mathbf{n}^{\text{solid}} \cdot \mathbf{u}^{\text{trp}} \tag{8.9}$$

or

$$n_x^{\text{solid}} u^{\text{solid}} + n_y^{\text{solid}} v^{\text{solid}} = n_x^{\text{solid}} u^{\text{trp}} + n_y^{\text{solid}} v^{\text{trp}} \tag{8.10}$$

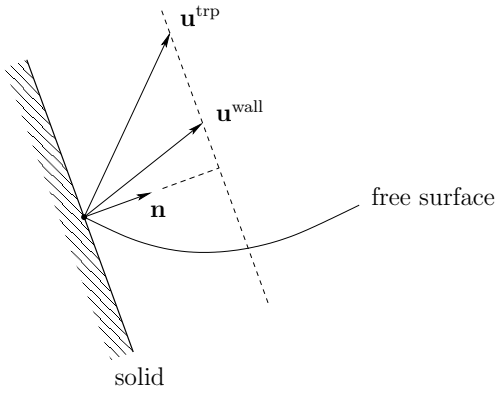

Figure 8.4: Moving solid boundary and a free surface.

where

$\mathbf{n}^{\text{solid}}$	normal of solid boundary at triple point position
$\mathbf{u}^{\text{solid}}$	velocity of solid boundary at triple point position
$\mathbf{u}^{\text{trp}}$	velocity of triple point

This equation states that the projection of the triple point velocity onto the solid boundary normal must equal projection of the solid boundary velocity onto the solid boundary normal. A graphical interpretation is sketched in figure 8.4 where valid triple point velocity vectors must end on the dashed line.

Thus one degree of freedom remains to be fixed in order to establish the velocity of the triple point. We have examined two different models for the determination of the remaining degree of freedom, namely 1) dynamic contact angle model, and 2) zero shear stress model. Each of these models are described in the following sections.

8.4.1 Dynamic contact angle model

A fluid which is at rest in a container will due to gravity and surface tension form a certain angle between the free surface and the container wall. If the container wall is not at rest the contact angle will in general be different from the steady state angle. These two situations are sketched in figure 8.5. The following is based on [43]. Using the definitions of velocities and normals in the figure we may define the *wetting speed* as

$$u_{\text{wet}} = \mathbf{t}^{\text{solid}} \cdot (\mathbf{u}^{\text{trp}} - \mathbf{u}^{\text{solid}}) \tag{8.11}$$

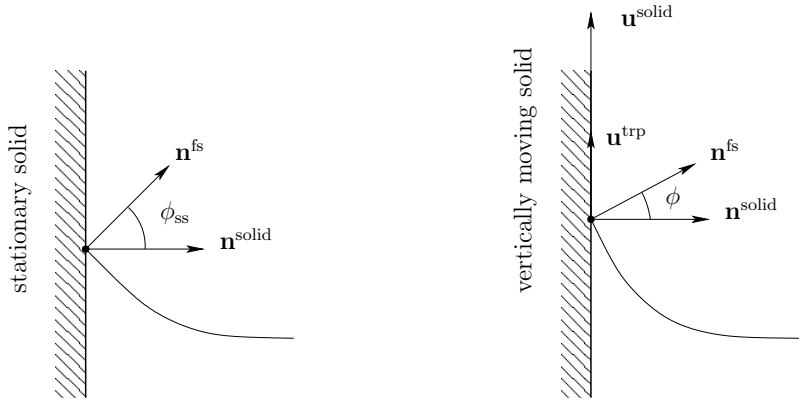

Figure 8.5: Left: Steady state contact angle. Right: Dynamic contact angle.

where

u_{wet}	wetting speed
$\mathbf{t}^{\text{solid}}$	tangent on solid boundary
$\mathbf{u}^{\text{solid}}$	velocity of solid boundary at triple point position
$\mathbf{u}^{\text{trp}}$	velocity of triple point

The wetting speed measures at what velocity relative to the solid boundary the *wetting point* (which is the same as the triple point) is moving. The following equation compares the contact angle (left hand side), the steady state angle, and a term due to wetting (right hand side)

$$\mathbf{n}^{\text{solid}} \cdot \mathbf{n}^{\text{fs}} = \cos\phi_{\text{ss}} - C_0 u_{\text{wet}} \tag{8.12}$$

where

$\mathbf{n}^{\text{fs}}$	normal on free surface at triple point
ϕ_{ss}	steady state contact angle
C_0	tuning parameter

It is seen that if the wetting speed is zero, then the contact angle must equal the steady state contact angle. If for some reason this is not fulfilled, then the wetting speed will be non-zero, which means that the triple point will be moving relative to the solid boundary. The wetting speed becomes proportional to the deviation of the contact angle from the steady state contact angle. This gives the name *dynamic contact angle model*, because the contact angle depends on the wetting speed.

Combining (8.11) and (8.12) gives

$$n_x^{\text{solid}} n_x^{\text{fs}} + n_y^{\text{solid}} n_y^{\text{fs}} = \cos\phi_{\text{ss}} - C_0 \left[n_x^{\text{solid}} (u^{\text{trp}} - u^{\text{solid}}) + n_y^{\text{solid}} (v^{\text{trp}} - v^{\text{solid}}) \right] \tag{8.13}$$

which together with (8.10) forms two equations for the two unknowns u^{trp} and v^{trp}. It is noted that these equations are linear and can be solved directly, giving

$$\begin{aligned} u^{\text{trp}} &= u^{\text{solid}} - n_y^{\text{solid}} C_1 \\ v^{\text{trp}} &= v^{\text{solid}} + n_x^{\text{solid}} C_1 \end{aligned} \tag{8.14}$$

where

$$C_1 = \frac{\cos\phi_{\text{ss}} - (n_x^{\text{solid}} n_x^{\text{fs}} + n_y^{\text{solid}} n_y^{\text{fs}})}{C_0}$$

(Note that a sign change of C_1 is needed in one end of the free surface, in order to prevent positive feedback. We do not comment on this technicality in the remaining part of this thesis.)

8.4.2 Zero shear stress model

The zero shear stress model is based on the idea that the triple point should be able to slide freely along the solid boundary. The ability to slide freely can be taken to the limit of no friction between the solid and the triple point. This translates into the property of zero shear stress at the triple point. That is, the velocity of the triple point must be such that the stress vector evaluated at the triple point has its shear component equal to zero, evaluated with respect to the normal of the solid boundary.

It is noted that this requirement is very similar to the shear part of the boundary conditions for a free surface (8.8, second equation). The only difference is that the normal $\mathbf{n}$ is now associated with the solid boundary instead of the free surface. Thus we write the equation for the zero shear stress model as

$$2\left(\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) n_x^{\text{solid}} n_y^{\text{solid}} + \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\left((n_y^{\text{solid}})^2 - (n_x^{\text{solid}})^2\right) = 0 \tag{8.15}$$

All quantities are evaluated at the triple point. This equation is combined with (8.10) in order to get two equations for the velocity components u^{trp} and v^{trp} of the triple point. It is noted that (8.15) and (8.10) form a linear set of equations.

This type of condition for the triple point has been investigated in [44], which also includes a treatment of other methods.

8.4.3 Comparison of methods

Before we end the presentation of modeling of triple points we give a short comparison of the two methods we have implemented. Beginning with the dynamic contact angle model we have

- model is based on kinematics

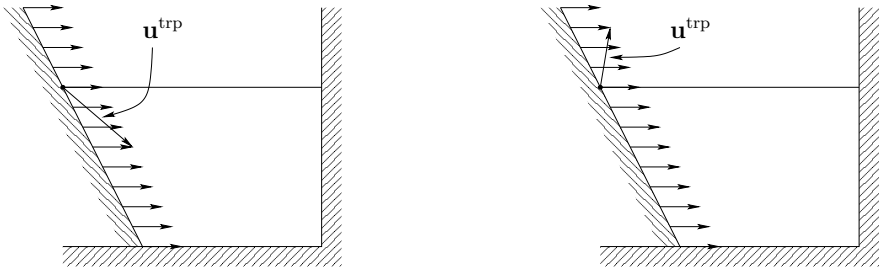

Figure 8.6: Snapshot at the initial time step. The interior of the fluid is at rest, and the inclined solid has non-zero horizontal velocity. Left: Dynamic contact angle model. Because the steady state constant angle is set to $\phi_{\mathrm{ss}} = \pi/2$ the triple point is moving downward the inclined solid boundary. After a short moment the correct contact angle is obtained, and the triple point then moves upward as the remaining part of the free surface. Right: Zero shear stress model. Because of the solid boundary velocity and the free slip condition the triple point is moving upward – already from the first time step.

- model has 2 user-defined parameters – the steady state constant angle ϕ_{ss} and the tuning parameter C_0

On the other hand we have for the zero shear stress model

- model is based on forces (stresses)
- model has no user-defined parameters

So the main difference is that the dynamic contact angle model takes its basis in the kinematics of the solid boundary and the shape of the free surface. This makes the method robust, but only if the value of the tuning parameter C_0 is selected appropriately (problem dependent). In fact a wrong value of C_0 can make a simulation blow up. On the other hand the zero shear stress model has no user-defined parameters, which makes the model easy to use. In figure 8.6 we have shown an example, that illustrates the basic difference between the two methods.

Chapter 9

Discretization of the Navier–Stokes equations

In this chapter we describe the discretization method used for the Navier–Stokes equations. We begin by discussing the overall strategy of the discretization method. Then we present how the spacial derivatives are approximated. After this we establish the discretization of the interior of the fluid domain and the boundary conditions. Finally, the time discretizations methods (an explicit and an implicit method) are presented.

We also discuss the special topics of remeshing and zero energy modes. The results obtained from the Navier–Stokes equations are found in the next chapter.

9.1 Discretization method

The overall discretization paradigm used in the work is known as the *semi-discretization* method. We are dealing with partial differential equations evolving in time over a 2D space. The idea of the semi-discretization method is to separate the discretization of time from the discretization of space.

A very useful feature of the procedure is that a partial differential equation, upon discretization, can be treated as a set of ordinary differential equations. This means that we can employ well-known schemes for time integration of ordinary differential equations. Furthermore it becomes relatively easy to change either the time integration method or the method for the spacial derivatives. Specifically we have taken advantage of this in order to investigate an explicit as well as an implicit time integration method.

Before we begin we recap a few things from the introduction, and reveal some more details about the discrete model.

Formulation The discretization is obtained from the strong form of the governing equations.

Coordinates We employ Lagrangian coordinates, which means that the discretization points are moving with the fluid flow.

Field variables All computational nodes carry the following information x, y (position), u, v (velocity components), ρ (density). Pressure p is not carried explicitly, but calculated when needed using the equation of state.

Data structure Nodes are contained in vectors, which are held by the various objects that make up a simulation problem. Thus, at each time step variables are picked from the local vectors and assembled into a *state vector* denoted by $\mathbf{s}$. The following degrees of freedom go into $\mathbf{s}$

- x, y, u, v, ρ for every node from the interior
- x, y for every node on free surfaces, except the first and last node if triple points are assigned
- x, y for triple point nodes

The remaining degrees of freedom (x, y, u, v, ρ on solids and u, v, ρ on free surfaces and triple points) are determined by evaluation of the boundary conditions.

Time stepping The explicit time integration method only needs evaluation of the time derivative of the variables in the state vector $\mathbf{s}$. The implicit time integration method is more complicated because a set of nonlinear equations must be solved at every time step. The solution method is based on Newton-Raphson iterations, which need gradient information. For efficiency these gradients are evaluated analytically, and for this reason a few sections dealing with sensitivity analysis are present in this chapter.

It is believed that this short overview will be helpful for the reading of the rest of this chapter.

We begin by describing how to discretize space, followed by the discretization of the interior domain and the boundary conditions. After this we consider two different ways of how to actually enforce the necessary boundary conditions. Then the time integration routines are presented.

9.2 Moving Least Squares

We discretize space using the method of Moving Least Squares (MLS). This method belongs to the family of meshfree methods and can be derived from different starting points. For example, the method develops in the Reproducing Kernel method [17] which is a correction of the earlier Smoothed Particle Hydrodynamics (SPH). On the other hand one can interpret MLS as a generalization of the finite difference method [45].

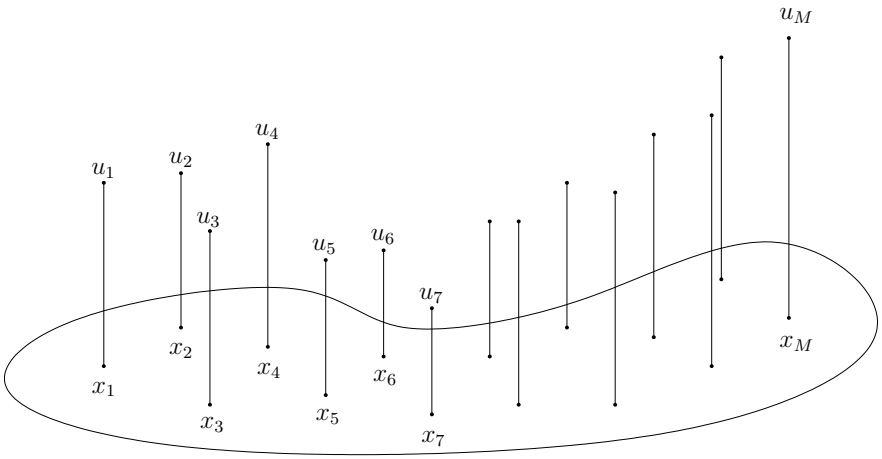

Figure 9.1: Data points and data values on domain Ω.

We follow a rather direct approach in which the idea is to approximate the true field by a polynomial surface. The surface is fitted to the data points in the sense of least squares, using only local information. That is, an approximating surface will be associated with each computational point.

In the following we derive the method using two different starting points 1) surface fitting, and 2) Taylor series.

9.2.1 Surface fitting

In figure 9.1 a sketch of data points $\mathbf{x}_i = (x_i, y_i)$ and data values u_i of some field variable is shown. Our goal is to achieve approximations of the derivatives of u at the data points $\mathbf{x}_i$. Of course this must be done using only the known values u_i defined at $\mathbf{x}_i$. The main idea is to approximate the true (unknown) field u using patches of polynomial surfaces. In fact one surface will be generated for each data point $\mathbf{x}_i$. This is sketched in figure 9.2.

The field variable u is approximated by

$$u^h(\mathbf{x}, \mathbf{x}^*) = \sum_{i=1}^{N} p_i(\mathbf{x} - \mathbf{x}^*) a_i(\mathbf{x}^*) \tag{9.1}$$

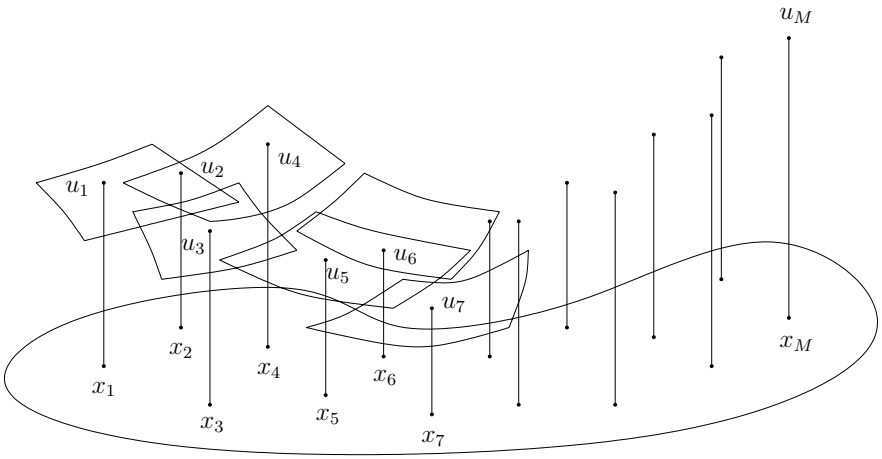

Figure 9.2: Polynomial surfaces approximating the true field u.

where

$\mathbf{x}$	point of evaluation
$\mathbf{x}^*$	point of expansion
p_i	basis functions for surface
a_i	coefficients for basis functions
N	number of basis functions

This expression allow us to calculate an approximation u^h at the point of evaluation $\mathbf{x}$. The expansion point $\mathbf{x}^*$ is used to offset the argument going into the basis functions p_i. We have used a nomial basis which is complete up to second order, that is we use

$$\begin{aligned} &p_1(\mathbf{x}) = 1 \qquad p_2(\mathbf{x}) = x \qquad p_3(\mathbf{x}) = y \\ &p_4(\mathbf{x}) = x^2 \qquad p_5(\mathbf{x}) = xy \qquad p_6(\mathbf{x}) = y^2 \end{aligned} \tag{9.2}$$

Note that special functions can be added to the list of basis functions. For example this has been exploited in applications for fracture mechanics [46]. However, we have not had that need and therefore use basis functions as given in (9.2).

In order to find the coefficients $\mathbf{a}$ the following functional is defined

$$\begin{aligned} J &= \sum_{j=1}^{M} W(h, \mathbf{x}_j - \mathbf{x}^*) \left[u^h(\mathbf{x}_j, \mathbf{x}^*) - u_j\right]^2 \\ &= \sum_{j=1}^{M} W(h, \mathbf{x}_j - \mathbf{x}^*) \left[\sum_{i=1}^{N} p_i(\mathbf{x}_j - \mathbf{x}^*) a_i(\mathbf{x}^*) - u_j\right]^2 \end{aligned} \tag{9.3}$$

where

W kernel (weight function)
h smoothing length
M number of data points

The functional sums the squared error between the known field values u_j and the values produced by the approximation (9.1). Each term in the sum is scaled by the factor W, which is known as the *kernel*. The kernel is a bell-shaped function with *compact support*. Having compact support ensures that the number of non-zero terms in (9.3) is modest, compared to the total number of data points. Furthermore it makes the functional J local.

We have used the bi-cubic spline kernel developed by Monaghan for SPH simulations [16]

$$W(h, \mathbf{x}-\mathbf{x}^*) = W(h,r) = \begin{cases} \frac{40}{7\pi h^2}\left(1 - 3/2q^2 + 3/4q^3\right) & \text{for } 0 \le q < 1 \\ \frac{40}{28\pi h^2}(2-q)^3 & \text{for } 1 \le q < 2 \\ 0 & \text{for } 2 \le q \end{cases} \tag{9.4}$$

where

$$r = \sqrt{(x-x^*)^2 + (y-y^*)^2}$$
$$q = \frac{2r}{h}$$

The *smoothing length* h denotes the radius of the circular support for W, and need not be constant for all data points $\mathbf{x}_i$. In figure 9.3 a plot of the kernel is shown. The functionality of the kernel is to adjust the significance of each term in (9.3). It is clear that a point outside the smoothing length will not contribute in (9.3). Also, points close to the point of expansion $\mathbf{x}^*$ will contribute more than points farther away. The support of a data point is called the *domain of dependence*.

The coefficients $\mathbf{a}$ are found by minimizing the functional J. This is done by satisfying the condition of stationarity

$$\begin{aligned} \frac{\partial J}{\partial a_k} &= \sum_{j=1}^{M} 2W(h, \mathbf{x}_j - \mathbf{x}^*) \left[\sum_{i=1}^{N} p_i(\mathbf{x}_j - \mathbf{x}^*) a_i(\mathbf{x}^*) - u_j \right] p_k(\mathbf{x}_j - \mathbf{x}^*) \\ &= 0 \end{aligned} \tag{9.5}$$

for $k = 1, 2, \ldots, N$ basis functions. Equation (9.5) can be rearranged into the matrix format

$$\mathbf{Ma} = \mathbf{b} \tag{9.6}$$

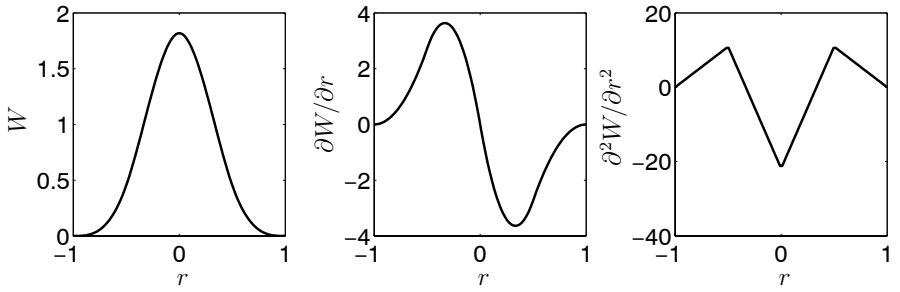

Figure 9.3: The bi-cubic kernel (9.4) as a function of r for smoothing length $h = 1$.

where, using the notation $W^j = W(h, \mathbf{x}_j - \mathbf{x}^*)$ and $p_i^j = p_i(\mathbf{x}_j - \mathbf{x}^*)$

$$\mathbf{M} = \begin{bmatrix} \sum W^j p_1^j p_1^j & \sum W^j p_2^j p_1^j & \sum W^j p_3^j p_1^j & \dots \\ \sum W^j p_1^j p_2^j & \sum W^j p_2^j p_2^j & \sum W^j p_3^j p_2^j & \dots \\ \sum W^j p_1^j p_3^j & \sum W^j p_2^j p_3^j & \sum W^j p_3^j p_3^j & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{9.7}$$

$$\mathbf{a} = \begin{pmatrix} a_1 & a_2 & a_3 & \dots \end{pmatrix}^{\mathrm{T}} \tag{9.8}$$

$$\mathbf{b} = \begin{pmatrix} \sum W^j p_1^j u_j & \sum W^j p_2^j u_j & \dots \end{pmatrix}^{\mathrm{T}} \tag{9.9}$$

where the summation index is $j = 1, 2, \dots, M$, with M being the number of data points. The matrix $\mathbf{M}$ is called the *matrix of moments* and has size N-by-N, with N being the number of basis functions.

Example in one dimension

Let us for a moment consider a small example defined in one dimension. We take the following basis functions

$$\begin{aligned} p_1 &= 1 \\ p_2 &= x \\ p_3 &= x^2 \end{aligned} \tag{9.10}$$

and define the following data points

$$\begin{aligned} \mathbf{x} &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} \\ \mathbf{u} &= \begin{pmatrix} 10 & 9 & 8 & 7 & 5 & 3 & 2 & 4 & 7 \end{pmatrix}^{\mathrm{T}} \end{aligned} \tag{9.11}$$

The smoothing length is set to $h = 2.5$. In figure 9.4 we have shown the approximated values u^h using two different points of expansion 1) $x^* = 2.2$, and

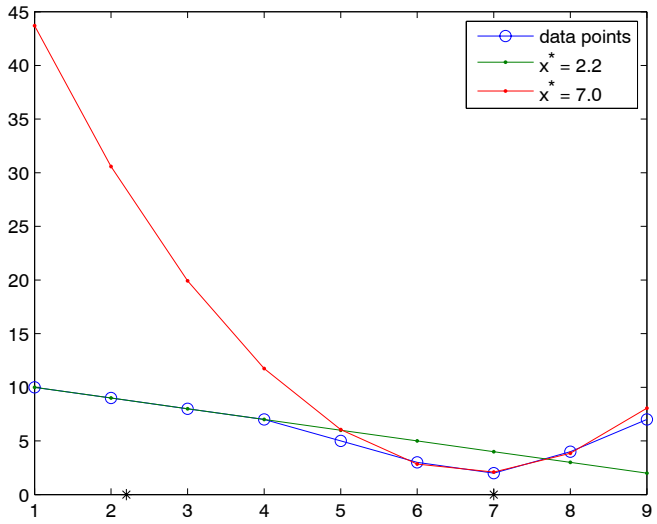

Figure 9.4: Least squares approximation based on two expansion points $x^* = 2.2$ and $x^* = 7$.

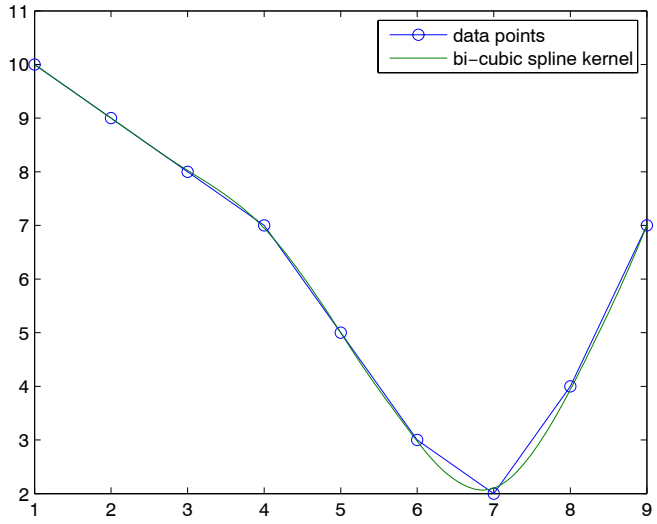

Figure 9.5: The approximated field u^h evaluated at 200 evenly distributed points from the interval $[1; 9]$. The kernel W is given by (9.4).

2) $x^* = 7$. From the figure we see that the linear part of the given data values are reproduced exactly. This happens because the basis functions span polynomials up to second order (and therefore also linear functions) and the data values within the smoothing length, measured from $x^* = 2.2$, are linearly distributed. On the other hand we see that the data values around $x = 7$ are not reproduced exactly. This happens because the data values within the smoothing length, measured from $x^* = 7$, cannot be described by a second order polynomial. Instead we have obtained an approximation by a least squares fit. We will return to this observation shortly.

Selection of expansion point

Now, in figure 9.4 we have two approximations based on two different expansion points. Clearly, each approximation is "good" in the proximity of the point of expansion, so the question is when do we need to change the expansion point? The answer is to make the expansion point and the evaluation point coincide at all times, so that $x^* = x$. This behavior explains the notion of *moving* least squares. In figure 9.5 we have shown a plot of the approximation u^h using moving least squares.

For the approximation shown in figure 9.5 we have used the kernel (9.4). The

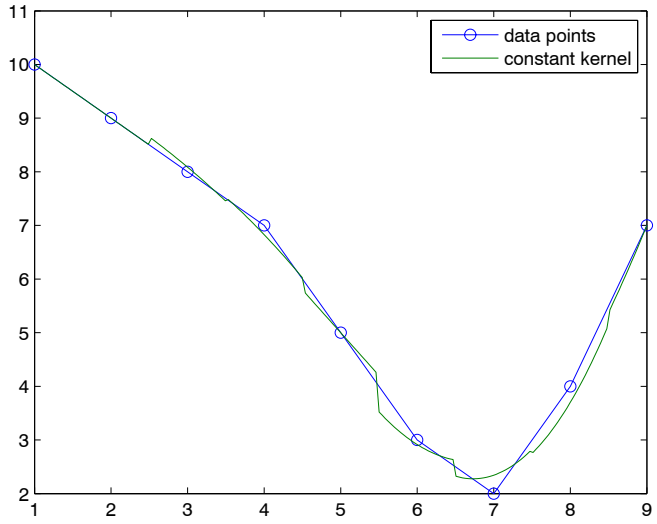

Figure 9.6: An approximation of the field u evaluated at 200 evenly distributed points from the interval $[1; 9]$. The kernel W is given by (9.12). Compare with figure 9.5.

approximation u^h becomes a smooth curve because the kernel has $W = 0$ on the boundary of the compact support, which allows a data point to enter or leave the domain of dependence quietly. For the sake of illustration we have plotted in figure 9.6 an approximation based on the kernel given by

$$W(h, r) = \begin{cases} 1 & \text{for } 0 \leq r \leq h \\ 0 & \text{for } h < r \end{cases} \tag{9.12}$$

In this case we get a discontinuous approximation of u, because data points contribute with full weight as soon as they enter the domain of dependence. This effect is not desirable.

Approximation or interpolation

In table 9.1 we have listed the MLS approximations evaluated at the locations of the given data points. As mentioned previously we see that some of the given data values are reproduced exactly by the MLS method, however, others are not. If the given field u can be spanned (locally) by the basis functions used in MLS method, then the data values will be reproduced, otherwise not. This means that MLS does not have the Kronecker delta property. Thus, in general

x	u	u^h
1	10	10.0000
2	9	9.0000
3	8	8.0254
4	7	6.9493
5	5	5.0000
6	3	2.9746
7	2	2.1015
8	4	3.9369
9	7	7.0000

Table 9.1: Given values of u and approximated values using MLS.

the MLS method produces *approximations* and not *interpolations*. This problem is common for many meshfree methods, and leads to difficulties when enforcing essential boundary conditions. Different methods such as penalization, Lagrange multiplier technique, or coupling with finite elements along the boundary has been proposed. We choose another more simple way of solving the problem.

The idea is to restrict the evaluation point x to coincide with a data point x_i. Since we have already restricted the expansion point x^* to the evaluation point x we now have $x = x^* = x_i$. With these restrictions a simple way of ensuring that $u_i^h = u(x_i)$ becomes available. We simply implement this equation directly in the MLS equation (9.6). In order to preserve a square matrix in (9.6) we cannot add a new equation, but must instead replace one of the existing rows. The obvious choice is the first row, so (9.6) then becomes

$$\begin{bmatrix} 1 & 0 & 0 & \dots \\ \sum W^j p_1^j p_2^j & \sum W^j p_2^j p_2^j & \sum W^j p_3^j p_2^j & \dots \\ \sum W^j p_1^j p_3^j & \sum W^j p_2^j p_3^j & \sum W^j p_3^j p_3^j & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} u(x_i) \\ \sum W^j p_2^j u_j \\ \sum W^j p_3^j u_j \\ \vdots \end{pmatrix} \quad (9.13)$$

Clearly, we can eliminate the first row by moving known values to the right hand side. The system will then be reduced from from N unknowns to $N-1$ unknowns. We get (remember that $p_1(x) = 1$)

$$\begin{bmatrix} \sum W^j p_2^j p_2^j & \sum W^j p_3^j p_2^j & \dots \\ \sum W^j p_2^j p_3^j & \sum W^j p_3^j p_3^j & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \sum W^j p_2^j \left(u_j - u(x_i)\right) \\ \sum W^j p_3^j \left(u_j - u(x_i)\right) \\ \vdots \end{pmatrix} \quad (9.14)$$

In figure 9.7 we have plotted the resulting polynomials for $x = x^* = x_7 = 7$ based on (9.6) and (9.14). In table 9.2 we have listed the coefficients $\mathbf{a}$. What we have obtained by (9.14) is that the Kronecker delta property has been satisfied at the point of evaluation of the MLS approximation. The price for this is that we can only generate the MLS approximation at a data point – since we need to know

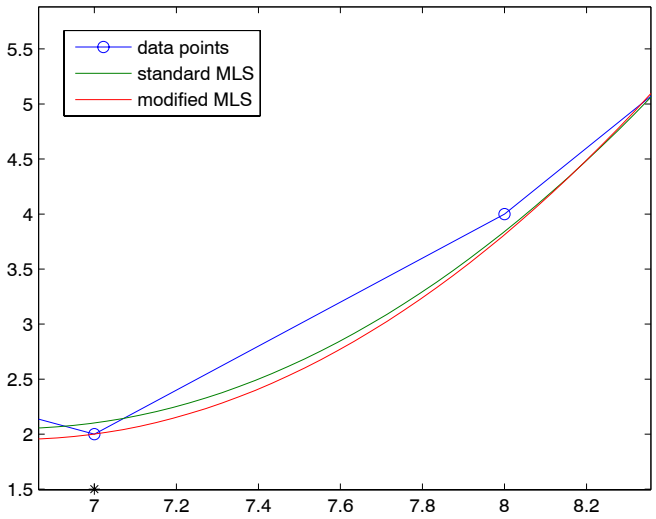

Figure 9.7: Comparison of the polynomials obtained by (9.6) and (9.14). The point of evaluation (which equals the point of expansion) is $x = 7$.

basis function	coefficients from (9.6)	coefficients from (9.14)
$p_1(x) = 1$	2.1015	2.0000
$p_2(x) = x$	0.5000	0.5000
$p_3(x) = x^2$	1.2389	1.3118

Table 9.2: Coefficient obtained by MLS based on (9.6) and (9.14).

the value of $u(x_i)$ in order to build the right hand side of (9.14). However, this is all we need for the discretization of partial differential equations as will be seen in section 9.5.

Therefore we adopt (9.14) as the equation on which we base the calculation of spacial derivatives. In order to make the presentation as clear as possible we write out the governing equation explicitly for our 2D application

$$\mathbf{M}\mathbf{a} = \mathbf{b} \tag{9.15}$$

where

$$\mathbf{M} = \begin{bmatrix} \sum W^i p_1^i p_1^i & \sum W^i p_1^i p_2^i & \sum W^i p_1^i p_3^i \\ \sum W^i p_2^i p_1^i & \sum W^i p_2^i p_2^i & \sum W^i p_2^i p_3^i \\ \sum W^i p_3^i p_1^i & \sum W^i p_3^i p_2^i & \sum W^i p_3^i p_3^i \\ \sum W^i p_4^i p_1^i & \sum W^i p_4^i p_2^i & \sum W^i p_4^i p_3^i \\ \sum W^i p_5^i p_1^i & \sum W^i p_5^i p_2^i & \sum W^i p_5^i p_3^i \end{bmatrix} \begin{bmatrix} \sum W^i p_1^i p_4^i & \sum W^i p_1^i p_5^i \\ \sum W^i p_2^i p_4^i & \sum W^i p_2^i p_5^i \\ \sum W^i p_3^i p_4^i & \sum W^i p_3^i p_5^i \\ \sum W^i p_4^i p_4^i & \sum W^i p_4^i p_5^i \\ \sum W^i p_5^i p_4^i & \sum W^i p_5^i p_5^i \end{bmatrix} \tag{9.16}$$

$$\mathbf{b} = \begin{pmatrix} \sum W^i \left(u_i - u^*\right) p_1^i \\ \sum W^i \left(u_i - u^*\right) p_2^i \\ \sum W^i \left(u_i - u^*\right) p_3^i \\ \sum W^i \left(u_i - u^*\right) p_4^i \\ \sum W^i \left(u_i - u^*\right) p_5^i \end{pmatrix} \tag{9.17}$$

using the notation $W^i = W(h, \mathbf{x}_i - \mathbf{x}^*)$ and $p_j^i = p_j(\mathbf{x}_i - \mathbf{x}^*)$. The summation index is $i = 1, 2, \ldots, M$ with M being the number of data points within the smoothing length of data point $\mathbf{x}^*$. Note that the basis functions are given by (this is a redefinition as compared with (9.2))

$$\begin{aligned} &p_1(\mathbf{x}) = x \qquad p_2(\mathbf{x}) = y \\ &p_3(\mathbf{x}) = x^2 \qquad p_4(\mathbf{x}) = xy \qquad p_5(\mathbf{x}) = y^2 \end{aligned} \tag{9.18}$$

This ends the section on the derivation of the MLS method based on surface fitting. In section 9.4 we show how to obtain spacial derivatives based on the coefficients $\mathbf{a}$ calculated from (9.15). But first we give an alternative derivation of the MLS method and a discussion on how the find nodes that are "close" to each other (i.e. within smoothing length). These two parts are found in the following sections.

9.2.2 Taylor expansion

In the previous section we derived the MLS method using data fitting as the starting point. In this section a derivation based on Taylor series is given. It

will be seen that MLS can also be considered as a generalization of the finite difference method.

Suppose a sufficiently smooth field u is given. The field can be expanded as a Taylor series from the expansion point x^*

$$\begin{aligned} u(\mathbf{x}) = u^* &+ \frac{\partial u}{\partial x}(x - x^*) + \frac{\partial u}{\partial y}(y - y^*) \\ &+ \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x - x^*)^2 + \frac{\partial^2 u}{\partial x \partial y}(x - x^*)(y - y^*) + \frac{1}{2}\frac{\partial^2 u}{\partial y^2}(y - y^*)^2 + \dots \end{aligned} \tag{9.19}$$

If we drop higher order terms (denoted by ellipsis above) and let $\mathbf{x} = \mathbf{x}_i$ we may write

$$\begin{aligned} u_i - u^* = &\frac{\partial u}{\partial x}(x_i - x^*) + \frac{\partial u}{\partial y}(y_i - y^*) \\ &+ \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x_i - x^*)^2 + \frac{\partial u}{\partial x \partial y}(x_i - x^*)(y_i - y^*) + \frac{1}{2}\frac{\partial^2 u}{\partial y^2}(y_i - y^*)^2 \end{aligned} \tag{9.20}$$

for $i = 1, 2, \dots, N$, where N is the number of nodes within the smoothing length of the node at $\mathbf{x}^*$. In the same way as for the MLS method we introduce a scaling of the equations using the kernel W given by (9.4). This gives using the matrix format

$$\mathbf{A}\mathbf{z} = \mathbf{b} \tag{9.21}$$

where

$$A = \begin{bmatrix} W^1\Delta x_1 & W^1\Delta y_1 & W^1\Delta x_1^2 & W^1\Delta x_1\Delta y_1 & W^1\Delta y_1^2 \\ W^2\Delta x_2 & W^2\Delta y_2 & W^2\Delta x_2^2 & W^2\Delta x_2\Delta y_2 & W^2\Delta y_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$z = \begin{pmatrix} \partial u/\partial x \\ \partial u/\partial y \\ 1/2 \cdot \partial^2 u/\partial x^2 \\ \partial^2 u/(\partial x \partial y) \\ 1/2 \cdot \partial^2 u/\partial y^2 \end{pmatrix}, \quad b = \begin{pmatrix} W^1(u_1 - u^*) \\ W^2(u_2 - u^*) \\ \vdots \end{pmatrix}$$

using the notation $\Delta x_j = x_j - x^*$ and $W^j = W(h, \mathbf{x}_j - \mathbf{x}^*)$. As in the case of MLS there will be more equations than unknowns in (9.21), so we solve for $\mathbf{z}$ by forming the normal equations

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{z} = \mathbf{A}^{\mathrm{T}}\mathbf{b} \tag{9.22}$$

which can be expanded into

$$\mathbf{A}^{\mathrm{T}}\mathbf{A} = \begin{bmatrix} \sum(W^i)^2\Delta x_i^2 & \sum(W^i)^2\Delta x_i\Delta y_i & \sum(W^i)^2\Delta x_i^3 \\ \sum(W^i)^2\Delta x_i\Delta y_i & \sum(W^i)^2\Delta y_i^2 & \sum(W^i)^2\Delta x_i^2\Delta y_i \\ \sum(W^i)^2\Delta x_i^3 & \sum(W^i)^2\Delta x_i^2\Delta y_i & \sum(W^i)^2\Delta x_i^4 \\ \sum(W^i)^2\Delta x_i^2\Delta y_i & \sum(W^i)^2\Delta x_i\Delta y_i^2 & \sum(W^i)^2\Delta x_i^3\Delta y_i \\ \sum(W^i)^2\Delta x_i\Delta y_i^2 & \sum(W^i)^2\Delta y_i^3 & \sum(W^i)^2\Delta x_i^2\Delta y_i^2 \end{bmatrix} \begin{bmatrix} \sum(W^i)^2\Delta x_i^2\Delta y_i & \sum(W^i)^2\Delta x_i\Delta y_i^2 \\ \sum(W^i)^2\Delta x_i\Delta y_i^2 & \sum(W^i)^2\Delta y_i^3 \\ \sum(W^i)^2\Delta x_i^3\Delta y_i & \sum(W^i)^2\Delta x_i^2\Delta y_i^2 \\ \sum(W^i)^2\Delta x_i^2\Delta y_i^2 & \sum(W^i)^2\Delta x_i\Delta y_i^3 \\ \sum(W^i)^2\Delta x_i\Delta y_i^3 & \sum(W^i)^2\Delta y_i^4 \end{bmatrix} \tag{9.23}$$

$$\mathbf{b} = \begin{pmatrix} \sum(W^i)^2\,(u_i - u^*)\,\Delta x_i \\ \sum(W^i)^2\,(u_i - u^*)\,\Delta y_i \\ \sum(W^i)^2\,(u_i - u^*)\,\Delta x_i^2 \\ \sum(W^i)^2\,(u_i - u^*)\,\Delta x_i\Delta y_i \\ \sum(W^i)^2\,(u_i - u^*)\,\Delta y_i^2 \end{pmatrix} \tag{9.24}$$

with the summation index $i = 1, 2, \ldots, N$, where N is the number of nodes within the smoothing length of $\mathbf{x}^*$. Comparing (9.22) with (9.15) we see that the equations are similar, except that the kernel has been squared in (9.22). This will of course change the weighting of each term, however, the two formulations are closely related.

The derivation in this section illustrates that MLS can also be interpreted as a generalization of the finite difference method.

9.3 Neighbor search

As already mentioned the MLS method belongs to the class of meshfree methods, meaning that the need for a well defined connectivity between the computational nodes is not necessary. However, we still need to keep track of the layout of nodes relative to each other. For example in order to calculate the coefficients $\mathbf{a}$ for a specific node j we need to know which nodes are within the compact support of node j. Because the position of nodes is updated according to the flow direction is it necessary to refresh this information as the simulation goes on. This process is known as *neighbor searching*.

The most simple strategy is to loop over all nodes when building the system of equations (9.15). The compact support of the kernel will automatically cancel the influence of nodes farther away than the smoothing length of the node under consideration. Clearly, this procedure has complexity N^2, where N is the total number of nodes. Because of this property we can only use this method for small problems.

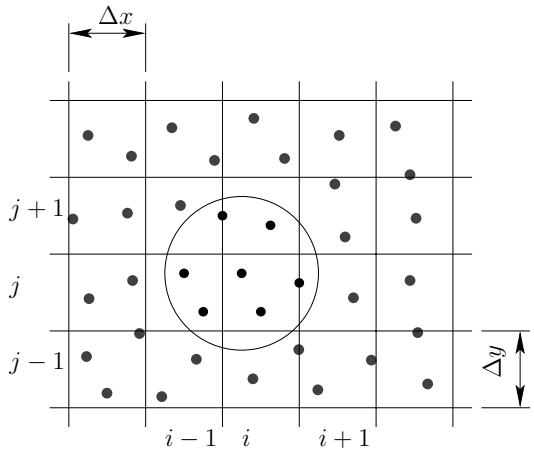

Figure 9.8: Neighbor search

A better way of searching for neighbors is to separate the detection of nodes within smoothing length (*interactions*) and the generation of equation (9.15). The neighbor search then becomes an individual step in the MLS procedure. In order to reduce the number of node comparisons we begin by partitioning the computational domain into cells on a rectangular grid, see figure 9.8. Each node is then mapped into the appropriate cell, using the following formula

$$\begin{aligned} i &= \texttt{floor}(x/\Delta x) \\ j &= \texttt{floor}(y/\Delta y) \end{aligned} \tag{9.25}$$

where

$\texttt{floor}(x)$	nearest integer less or equal to x
i	horizontal cell coordinate
j	vertical cell coordinate
x	x-coordinate of node
y	y-coordinate of node
Δx	width of cell
Δy	height of cell

If the (maximum) smoothing length is $h = h_{\max}$, then we may select $\Delta x = \Delta y = h_{\max}$ (If the compact support of the kernel is not circular, then Δx and Δy could be distinct.) This choice will guaranty that a node belonging to cell (i, j) can have neighbors in the nine cells $(i-1, j-1)$, $(i-1, j)$, $(i-1, j+1)$, $(i, j-1)$, (i, j), $(i, j+1)$, $(i+1, j-1)$, $(i+1, j)$, and $(i+1, j+1)$ only. Thus the number

of comparisons is significantly reduced. In fact the complexity of this algorithm is $O(N)$, with N being the total number of nodes.

When a node, say node i, is found to be within the smoothing length of node j, then we have basically two options for storing the information. Either we can 1) add an entry in a list of *interaction pairs*, or 2) append the information in a *interaction list* of node i (interaction with j is added) and a interaction list of node j (interaction with i is added). It is seen that the latter choice will store the same information twice. However, despite this drawback on memory requirements and operations, we choose this option because this data structure simplifies the creation of equation (9.15). The interaction pair data structure is useful for e.g. SPH which does not have an explicitly formed equation like (9.15), but it is not convenient for this application.

Once all nodes are mapped into cells the interactions are found by a single sweep through all nodes. Starting by node 1 we detect the nodes within its smoothing length, and put the information in the interaction list belonging to node 1. Also, we update the interaction lists of node 1's neighbors. After this we can safely remove node 1 from its cell, because all possible interactions with node 1 are detected and stored in the relevant interaction lists. This means that the interaction between two nodes is not *detected* twice. This performance optimization is exploited in the simulation program.

Note that this procedure is only valid if "node i is within smoothing length of node j" implies that "node j is within the smoothing length of node i". This is obviously the case if all nodes have the same smoothing length. If this is not the case we may use the following averaging

$$h = \frac{h_i + h_j}{2} \tag{9.26}$$

where h_i is the smoothing length of node i and h_j is the smoothing length of node j. So the criteria for interaction becomes

$$\sqrt{(x_i - x_j)^2 + (y_i + y_j)^2} < \frac{h_i + h_j}{2} \tag{9.27}$$

This is the condition we have implemented in the program.

Finally it should be mentioned that it might be a good a idea to cache some of the values that are (or can be) calculated along the interaction search. From (9.15) is it seen that allot of evaluations of the kernel function are needed. These values can be calculated once and stored in a list in a similar way as the interaction information. A balance between memory requirements and gain in computational speed is decisive for how many values that should be stored.

9.4 Spacial derivatives

In section 9.2.1 we saw how to obtain the coefficients for a polynomial surface approximating the given data values at each data point. Once these coefficients

have been calculated we can obtain approximations of the spacial derivatives by differentiating (9.1) which is restated here

$$u^h(\mathbf{x}, \mathbf{x}^*) = \sum_{i=1}^{N} p_i(\mathbf{x} - \mathbf{x}^*) a_i(\mathbf{x}^*) \tag{9.28}$$

Taking the derivative with respect to x gives

$$\frac{\partial u^h(\mathbf{x}, \mathbf{x}^*)}{\partial x} = \sum_{i=1}^{N} \frac{\partial p_i(\mathbf{x} - \mathbf{x}^*)}{\partial x} a_i(\mathbf{x}^*) \tag{9.29}$$

If we substitute the expressions for the basis functions (9.18) we get

$$\frac{\partial u^h}{\partial x} = a_1 + 2(x - x^*)a_3 + (y - y^*)a_4 \tag{9.30}$$

Since we want the derivative at the data point, we evaluate the expression above for $x = x^*$. This gives the final result

$$\frac{\partial u^h}{\partial x} = a_1 \tag{9.31}$$

In a similar way we can obtain partial derivatives with respect to y and higher order derivatives. Below we list the relevant derivatives needed for the discretization of the Navier-Stokes equations. For each node we get

$$\begin{array}{lll}
\dfrac{\partial u}{\partial x} = \mathbf{e}_1\mathbf{M}^{-1}\mathbf{b}^u & \dfrac{\partial u}{\partial y} = \mathbf{e}_2\mathbf{M}^{-1}\mathbf{b}^u & \\
\dfrac{\partial^2 u}{\partial x^2} = 2\mathbf{e}_3\mathbf{M}^{-1}\mathbf{b}^u & \dfrac{\partial^2 u}{\partial x \partial y} = \mathbf{e}_4\mathbf{M}^{-1}\mathbf{b}^u & \dfrac{\partial^2 u}{\partial y^2} = 2\mathbf{e}_5\mathbf{M}^{-1}\mathbf{b}^u \\
\dfrac{\partial v}{\partial x} = \mathbf{e}_1\mathbf{M}^{-1}\mathbf{b}^v & \dfrac{\partial v}{\partial y} = \mathbf{e}_2\mathbf{M}^{-1}\mathbf{b}^v & \\
\dfrac{\partial^2 v}{\partial x^2} = 2\mathbf{e}_3\mathbf{M}^{-1}\mathbf{b}^v & \dfrac{\partial^2 v}{\partial x \partial y} = \mathbf{e}_4\mathbf{M}^{-1}\mathbf{b}^v & \dfrac{\partial^2 v}{\partial y^2} = 2\mathbf{e}_5\mathbf{M}^{-1}\mathbf{b}^v \\
\dfrac{\partial \rho}{\partial x} = \mathbf{e}_1\mathbf{M}^{-1}\mathbf{b}^\rho & \dfrac{\partial \rho}{\partial y} = \mathbf{e}_2\mathbf{M}^{-1}\mathbf{b}^\rho &
\end{array} \tag{9.32}$$

where $\mathbf{e}_j$ denotes the j-th standard basis vector and $\mathbf{M}$ is defined as in (9.16). Note that $\mathbf{M}^{-1}$ is common for all derivatives at a specific node, so only a single matrix factorization is needed. Also $\mathbf{M}$ is symmetric and positive definite, so the Cholesky factorization can be used. The right hand sides $\mathbf{b}^u$, $\mathbf{b}^v$, and $\mathbf{b}^\rho$ are defined as in (9.17) based on the field variables u, v, and ρ respectively. The notation using $\mathbf{e}_j$ is only for illustrating purposes, showing how each entry in the solution vector $\mathbf{a}$ from (9.15) corresponds to a certain derivative.

9.4.1 Sensitivity analysis

In the previous sections of this chapter we have seen how to obtain approximations of the spacial derivatives based on the given field values at the discrete computational nodes. In the following we will need also the *sensitivity* of the spacial derivatives. For example we would like to known how $\partial u_i/\partial x$ changes when either the location or the field value at some node j changes. It is clear that there will be no change if node j is outside the smoothing length of node i. Therefore, in the following we assume that node i and node j are within the smoothing length of each other.

The spacial derivatives of a field variable are given by (9.15) (except for a factor of 2 on the second order derivatives with respect to x and y), which is an equation of the form

$$\mathbf{M}_i \mathbf{d}_i = \mathbf{b}_i \tag{9.33}$$

We have included the subscript $_i$ to indicate that the equation is related to node i. Taking the derivative of this equation with respect to x_j (the x-position of node j) gives

$$\frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i + \mathbf{M}_i \frac{\partial \mathbf{d}_i}{\partial x_j} = \frac{\partial \mathbf{b}_i}{\partial x_j} \tag{9.34}$$

Rearranging this gives

$$\frac{\partial \mathbf{d}_i}{\partial x_j} = \mathbf{M}^{-1}\left(\frac{\partial \mathbf{b}_i}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i\right) \tag{9.35}$$

where $\mathbf{d}_i = \mathbf{M}_i^{-1}\mathbf{b}_i$. The derivative of $\mathbf{M}$ is available by explicit differentiation. Two cases exist 1) $i = j$, and 2) $i \neq j$. The first case occurs when sensitivities with respect to changes in the node itself are needed. The first entry for the case $i = j$ is listed below

$$\begin{aligned}\left(\frac{\partial \mathbf{M}_i}{\partial x_i}\right)_{11} &= \sum_{k=1}^{N}\left(\frac{\partial W(h,\mathbf{x}_k-\mathbf{x}_i)}{\partial x}\frac{\partial (x_k-x_i)}{\partial x_i}p_1(\mathbf{x}_k-\mathbf{x}_i)p_1(\mathbf{x}_k-\mathbf{x}_i)\right.\\ &\qquad \left. +2W(h,\mathbf{x}_k-\mathbf{x}_i)\frac{\partial p_1(\mathbf{x}_k-\mathbf{x}_i)}{\partial x}\frac{\partial (x_k-x_i)}{\partial x_i}\right)\\ &= -\sum_{\substack{k=1\\k\neq i}}^{N}\left(\frac{\partial W(h,\mathbf{x}_k-\mathbf{x}_i)}{\partial x}p_1(\mathbf{x}_k-\mathbf{x}_i)p_1(\mathbf{x}_k-\mathbf{x}_i)\right.\\ &\qquad \left. +2W(h,\mathbf{x}_k-\mathbf{x}_i)\frac{\partial p_1(\mathbf{x}_k-\mathbf{x}_i)}{\partial x}\right)\end{aligned} \tag{9.36}$$

Note that we have excluded the term corresponding to $k = i$ from the summation above. This is due to the fact that this term is identically zero, since

1. $p_n(\mathbf{0}) = 0$ for all $n = 1, 2, \ldots, 5$
2. $\partial W(h, \mathbf{0})/\partial x = 0$

The second case $i \neq j$ is more simple because most of the terms in the summation become zero. The cancellation of terms is due to

1. $\partial(x_k - x_i)/\partial x_j$ is only non-zero for $k = j$ in which case it evaluates to 1

So in total we get for $i \neq j$

$$\begin{aligned}\left(\frac{\partial \mathbf{M}_i}{\partial x_j}\right)_{11} =& \frac{\partial W(h, \mathbf{x}_j - \mathbf{x}_i)}{\partial x} p_1(\mathbf{x}_j - \mathbf{x}_i) p_1(\mathbf{x}_j - \mathbf{x}_i) \\ &+ 2W(h, \mathbf{x}_j - \mathbf{x}_i) \frac{\partial p_1(\mathbf{x}_j - \mathbf{x}_i)}{\partial x}\end{aligned} \tag{9.37}$$

The remaining entries in the matrix $\partial \mathbf{M}/\partial x_j$ are found in a similar way. The derivative of the right hand side $\partial \mathbf{b}/\partial x_j$ may also be obtained using the same computations already shown. Again two cases develop. Below we show the first entry for $i = j$, supposing that the u-field is under consideration

$$\begin{aligned}\left(\frac{\partial \mathbf{b}_i^u}{\partial x_i}\right)_1 = -\sum_{\substack{k=1 \\ k \neq i}}^{M} \Bigg(& \frac{\partial W(h, \mathbf{x}_k - \mathbf{x}_i)}{\partial x} p_1(\mathbf{x}_k - \mathbf{x}_i) \\ & + W(h, \mathbf{x}_k - \mathbf{x}_i) \frac{\partial p_1(\mathbf{x}_k - \mathbf{x}_i)}{\partial x} \Bigg) (u_k - u_i)\end{aligned} \tag{9.38}$$

and for $i \neq j$

$$\begin{aligned}\left(\frac{\partial \mathbf{b}_i^u}{\partial x_j}\right)_1 = & \Bigg(\frac{\partial W(h, \mathbf{x}_j - \mathbf{x}_i)}{\partial x} p_1(\mathbf{x}_j - \mathbf{x}_i) \\ & + W(h, \mathbf{x}_j - \mathbf{x}_i) \frac{\partial p_1(\mathbf{x}_j - \mathbf{x}_i)}{\partial x} \Bigg) (u_j - u_i)\end{aligned} \tag{9.39}$$

Having established the derivatives of the matrix (9.16) and the right hand side (9.17) it is now possible to compose equation (9.35). The sensitivities can then be found from the following expressions

$$\begin{aligned}\frac{\partial}{\partial x_j}\left(\frac{\partial u}{\partial x}\right)_i &= \mathbf{e}_1 \mathbf{M}_i^{-1} \left(\frac{\partial \mathbf{b}_i^u}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i\right) \\ \frac{\partial}{\partial x_j}\left(\frac{\partial u}{\partial y}\right)_i &= \mathbf{e}_2 \mathbf{M}_i^{-1} \left(\frac{\partial \mathbf{b}_i^u}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i\right) \\ \frac{\partial}{\partial x_j}\left(\frac{\partial^2 u}{\partial x^2}\right)_i &= 2\mathbf{e}_3 \mathbf{M}_i^{-1} \left(\frac{\partial \mathbf{b}_i^u}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_i}\mathbf{d}_i\right) \\ \frac{\partial}{\partial x_j}\left(\frac{\partial^2 u}{\partial x \partial y}\right)_i &= \mathbf{e}_4 \mathbf{M}_i^{-1} \left(\frac{\partial \mathbf{b}_i^u}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i\right) \\ \frac{\partial}{\partial x_j}\left(\frac{\partial^2 u}{\partial x^2}\right)_i &= 2\mathbf{e}_5 \mathbf{M}_i^{-1} \left(\frac{\partial \mathbf{b}_i^u}{\partial x_j} - \frac{\partial \mathbf{M}_i}{\partial x_j}\mathbf{d}_i\right)\end{aligned} \tag{9.40}$$

where $\mathbf{e}_k$ is the k-th standard basis vector. In practice the sensitivities above are calculated simultaneously as a vector operation, and not one by one. The same kind of expressions develop for the sensitivities of $\partial v/\partial x$, ..., $\partial v^2/\partial y^2$ and $\partial\rho/\partial x$, $\partial\rho/\partial y$ with respect to x_j as well as with respect to y_j. Note the factor of 2 that must be added to the second order derivatives with respect to x and y (refer to (9.32)).

Similar computations can by carried out for the sensitivities with respect to field variables, say for example u_j. In this case things are, however, more simple because the matrix $\mathbf{M}_i$ of (9.33) does not depend on the field variables. Therefore we simply get

$$\mathbf{M}_i \frac{\partial \mathbf{d}_i}{\partial u_j} = \frac{\partial \mathbf{b}_i}{\partial u_j} \tag{9.41}$$

The right hand side of this equation is also very simple, and can be described by a few special cases.

First it is noted that the right hand side of (9.17) depends on only u, v, and ρ according to which field variable we are dealing with (it also depends on x and y but these quantities are constant in this context). This means the right hand side will become zero if we look for the sensitivity of e.g. u with respect to v. If the right hand side is zero, so are the corresponding sensitivities. Therefore the only nontrivial cases are sensitivities of u_i w.r.t. u_j, sensitivities of v_i w.r.t. v_j, and sensitivities of ρ_i w.r.t. ρ_j.

Looking at (9.17) we see that the right hand side of (9.41) does not depend on the field value. In fact if we consider the u-field the first entry in the right hand side becomes, for $i = j$

$$\left(\frac{\partial \mathbf{b}_i^u}{\partial u_i}\right)_1 = -\sum_{\substack{k=1 \\ k\neq i}} W(h, \mathbf{x}_k - \mathbf{x}_i) p_1(\mathbf{x}_k - \mathbf{x}_i) \tag{9.42}$$

and for $i \neq j$ we get

$$\left(\frac{\partial \mathbf{b}_i^u}{\partial u_j}\right)_1 = W(h, \mathbf{x}_j - \mathbf{x}_i) p_1(\mathbf{x}_j - \mathbf{x}_i) \tag{9.43}$$

Clearly, partial derivatives of $\mathbf{b}_i^u$ with respect to u does not depend on u itself. The conclusion is that since neither the left hand side nor the right hand side of (9.41) depends on the field value, so doesn't the sensitivities. Thus we can calculate the sensitivities with respect to changes in any field value by a single

equation

$$\begin{aligned}
\frac{\partial}{\partial(\cdot)_j}\left(\frac{\partial(\cdot)}{\partial x}\right)_i &= \mathbf{e}_1\mathbf{M}_i^{-1}\frac{\partial \mathbf{b}_i^u}{\partial u_j} \\
\frac{\partial}{\partial(\cdot)_j}\left(\frac{\partial(\cdot)}{\partial y}\right)_i &= \mathbf{e}_2\mathbf{M}_i^{-1}\frac{\partial \mathbf{b}_i^u}{\partial u_j} \\
\frac{\partial}{\partial(\cdot)_j}\left(\frac{\partial^2(\cdot)}{\partial x^2}\right)_i &= 2\mathbf{e}_3\mathbf{M}_i^{-1}\frac{\partial \mathbf{b}_i^u}{\partial u_j} \\
\frac{\partial}{\partial(\cdot)_j}\left(\frac{\partial^2(\cdot)}{\partial x\partial y}\right)_i &= \mathbf{e}_4\mathbf{M}_i^{-1}\frac{\partial \mathbf{b}_i^u}{\partial u_j} \\
\frac{\partial}{\partial(\cdot)_j}\left(\frac{\partial^2(\cdot)}{\partial x^2}\right)_i &= 2\mathbf{e}_5\mathbf{M}_i^{-1}\frac{\partial \mathbf{b}_i^u}{\partial u_j}
\end{aligned} \tag{9.44}$$

where $(\cdot)$ can be any of u, v, and ρ for the same right hand side. This computational saving is taken into account in the computer implementation.

It is also noted that all sensitivities are obtained by reusing a single factorization of the matrix $\mathbf{M}$ and substitution of the right hand sides as needed.

9.5 Governing equations

Having established the discrete approximations of the spacial derivatives we are now able to discretize the right hand side of the Navier–Stokes equations (6.5) and the continuity equation (6.7). As mentioned in the introduction of this chapter the computational nodes carry density ρ but not pressure p. The distinction reflects that we do not calculate spacial derivatives of pressure explicitly. Instead we use the following relationship

$$\begin{aligned}
\frac{\partial p}{\partial x} &= \frac{\mathrm{d}p}{\mathrm{d}\rho}\frac{\partial \rho}{\partial x} \\
\frac{\partial p}{\partial y} &= \frac{\mathrm{d}p}{\mathrm{d}\rho}\frac{\partial \rho}{\partial y}
\end{aligned} \tag{9.45}$$

The derivative of p with respect to ρ is obtained from the equation of state (refer to section 6.2).

The positional update (6.6) of the nodes is discretized by

$$\begin{aligned}
\frac{\mathrm{d}x_i}{\mathrm{d}t} &= u_i \\
\frac{\mathrm{d}y_i}{\mathrm{d}t} &= v_i
\end{aligned} \tag{9.46}$$

We discretize the Navier–Stokes equations (6.5) in the following way

$$\begin{aligned}\frac{\mathrm{d}u_i}{\mathrm{d}t} &= \frac{1}{\rho_i}\left[-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\partial\rho_i}{\partial x} + \mu\left(\frac{4}{3}\frac{\partial^2 u_i}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v_i}{\partial x\partial y} + \frac{\partial^2 u_i}{\partial y^2}\right)\right] \\ \frac{\mathrm{d}v_i}{\mathrm{d}t} &= \frac{1}{\rho_i}\left[-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\partial\rho_i}{\partial x} + \mu\left(\frac{4}{3}\frac{\partial^2 u_i}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v_i}{\partial x\partial y} + \frac{\partial^2 u_i}{\partial y^2}\right)\right]\end{aligned} \tag{9.47}$$

and the continuity equation (6.7) by

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho_i\left(\frac{\partial u_i}{\partial x} + \frac{\partial v_i}{\partial y}\right) \tag{9.48}$$

where spacial derivatives of u, v, and ρ are evaluated using (9.32). The subscript $_i$ indicates that the expressions above are evaluated for every node i in the interior of the fluid domain.

The left hand side will be discretized later on in sections 9.7.2 and 9.7.3. This is an illustration of the separation of the spacial discretization and the time discretization of the semi-discretization technique.

9.5.1 Gradients of the governing equations

Later on in section 9.7.3 we will need the gradients of the governing equations. It seems natural to include the expressions here. For the positional update (9.46) we get

$$\begin{aligned}&\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right) = 0 \quad \frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right) = 0 \\ &\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right) = \delta_{ij} \quad \frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right) = 0 \quad \frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right) = 0\end{aligned} \tag{9.49}$$

and

$$\begin{aligned}&\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right) = 0 \quad \frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right) = 0 \\ &\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right) = 0 \quad \frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right) = \delta_{ij} \quad \frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right) = 0\end{aligned} \tag{9.50}$$

For the Navier–Stokes equations (9.47) we get

$$\begin{aligned}\frac{\partial}{\partial(\cdot)_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right) = \frac{1}{\rho_i}\Bigg[&-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial\rho_i}{\partial x}\right) \\ &+\mu\left(\frac{4}{3}\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial^2 u_i}{\partial x^2}\right) + \frac{1}{3}\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial^2 v_i}{\partial x\partial y}\right) + \frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial^2 u_i}{\partial y^2}\right)\right)\Bigg]\end{aligned} \tag{9.51}$$

where $(\cdot)$ denotes either of x, y, u, or v. With respect to ρ_j we get

$$\begin{aligned}\frac{\partial}{\partial\rho_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right) = \frac{1}{\rho_i}\Bigg[&-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\mathrm{d}}{\mathrm{d}\rho_j}\left(\frac{\partial\rho_i}{\partial x}\right)\\ &+\mu\left(\frac{4}{3}\frac{\mathrm{d}}{\mathrm{d}\rho_j}\left(\frac{\partial^2 u_i}{\partial x^2}\right)+\frac{1}{3}\frac{\mathrm{d}}{\mathrm{d}\rho_j}\left(\frac{\partial^2 v_i}{\partial x\partial y}\right)+\frac{\partial}{\partial\rho_j}\left(\frac{\partial^2 u_i}{\partial y^2}\right)\right)\Bigg]\\ &-\frac{1}{(\rho_i)^2}\delta_{ij}\left[-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\partial\rho_i}{\partial x}+\mu\left(\frac{4}{3}\frac{\partial^2 u_i}{\partial x^2}+\frac{1}{3}\frac{\partial^2 v_i}{\partial x\partial y}+\frac{\partial^2 u_i}{\partial y^2}\right)\right]\end{aligned} \tag{9.52}$$

Similar expressions can be derived for the v-momentum equation. We do not display those results here. Finally we get for the continuity equation (9.48)

$$\frac{\partial}{\partial(\cdot)_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right) = -\rho_i\left[\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial u}{\partial x}\right)+\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(\frac{\partial v}{\partial y}\right)\right] \tag{9.53}$$

where $(\cdot)$ is either of x, y, u, or v. With respect to ρ_j we get

$$\frac{\partial}{\partial\rho_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right) = -\rho_i\left[\frac{\mathrm{d}}{\mathrm{d}\rho_j}\left(\frac{\partial u_i}{\partial x}\right)+\frac{\mathrm{d}}{\mathrm{d}\rho_j}\left(\frac{\partial v_i}{\partial y}\right)\right]-\delta_{ij}\left[\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right] \tag{9.54}$$

Note that we have taken the *total derivative* of the spacial derivatives on the right hand sides above. This is needed because the partial derivatives of section 9.4.1 does not take all dependencies into account by themselves. The extra dependencies arise from the boundary conditions as will be seen in section 9.7.3. Thus, we postpone the definition of e.g. $\mathrm{d}(\partial u_i/\partial x)/\mathrm{d}\rho_j$ and similar terms to that section.

9.6 Boundary conditions

In this section we describe the implementation of the boundary conditions in the discrete model. The treatment of boundary conditions is very important for any mathematical model. One could say that the "natural" domain for a differential equation spreads infinitely in the spacial directions and one half-plane in the time direction (either forward in time or backward in time). However, the typical situation is that the differential equation applies only to some subdomain. Thus we have to truncate the "natural" infinite domain – this is done by specifying appropriate boundary conditions.

The boundary conditions may be divided into two groups 1) prescribed boundary values, and 2) derived boundary values. An example of the first category is the velocity components at the solid boundaries. Since we only consider solids with prescribed kinematics, these values can be calculated directly at any moment in time. An example of the second category is the velocity components at a free surface. These values depend on the interior of the fluid domain, and are calculated as the solution to certain equations that must be satisfied on the free surface boundary. In the following section we derive these equations for the discrete model.

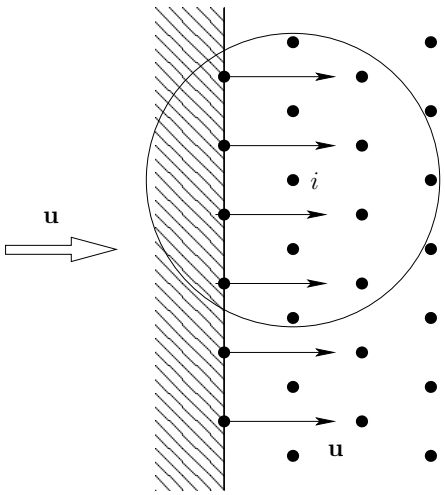

Figure 9.9: Boundary conditions for a solid. The circle indicates the support of node i.

9.6.1 Residual equations

This section deals with equations that must be satisfied on the boundary for the discrete model. We begin with solid boundaries, then the boundary condition for density, free surfaces, and finally triple points.

In each case the governing equations for the values on the boundaries are developed and expressed in residual form. This form is useful when the equations are nonlinear, since then Newton-Raphson iterations can be applied in order to make the residual zero and find the solution. Even if the equations are linear we keep the residual formulation. In this case the solution is found after just one iteration, and virtually no extra work is induced from the Newton-Raphson algorithm. We have collected the gradients needed for the Newton-Raphson method in section 9.6.2.

Solid boundaries

At a solid boundary we employ the no-slip condition. This means that the velocity field of a fluid particle at the boundary must equal the velocity of the solid boundary at that location. In figure 9.9 we have sketched the situation for a moving solid boundary. We only consider solids with prescribed kinematics in this thesis. Therefore the velocity components indicated in figure 9.9 can be calculated at all times, using the formulas (7.6). The figure also shows how the information is transfered into the interior of the domain, through the domain of dependence of the fluid nodes. The values of u and v on the boundary will affect

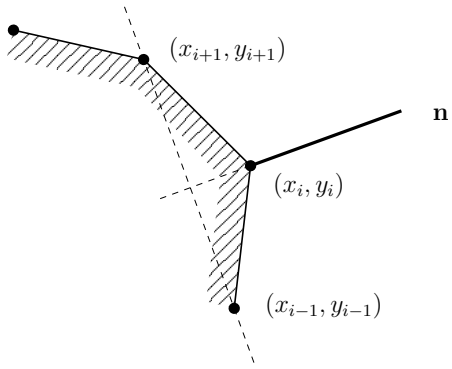

Figure 9.10: Calculation of normals on a solid boundary (sketch).

the spacial derivatives of velocity at the fluid nodes.

The density boundary condition for solid boundaries is described in the following section. In relation to this it is mentioned that the acceleration of a point on a solid boundary is given by the kinematic transformations (7.6).

Also, we calculate the normal of a solid boundary in the following way

$$\begin{aligned} n_x^i &= -\frac{y_{i+1} - y_{i-1}}{L} \\ n_y^i &= \frac{x_{i+1} - x_{i-1}}{L} \end{aligned} \tag{9.55}$$

where

$$\begin{aligned} n_x^i &\quad x\text{-component of normal at node } i \\ n_y^i &\quad y\text{-component of normal at node } i \\ L &= \sqrt{(x_{i+1} - x_{i-1})^2 + (y_{i+1} - y_{i-1})^2} \end{aligned}$$

In figure 9.10 we have shown the calculation of normals graphically. It should be noted that (9.55) is only valid if the nodal spacing at the solid boundary is even. This is in fact the case, as will be explained in section 9.8.1.

Density boundary condition

If we look at the Navier–Stokes equations as given in (9.47)-(9.48) we see that we need spacial derivatives of density. However, the density is not directly available on the boundary (solids, free surfaces, and triple points). As already mentioned in section 8.2 there are basically two options when calculating spacial derivatives of density 1) do not include nodes on the boundary (because the value of density is not available here), or 2) establish the value of density at the boundaries. The

first choice is common if one is using a discretization method with staggered grids for velocity and density. In this case one would arrange the grids so that the velocity nodes coincides with the solid boundaries. Although we do not have staggered grids we have tested method 1 and found it to be working OK. We will not give details on this, however, since we anyhow prefer to calculate density on the boundaries (method 2) for the following reasons

1. The stress vector on the solid boundaries is needed in order to calculate e.g. the load carrying capacity of a bearing

2. It takes extra work to have different domains of dependence for different field variables

The first argument is that the stress vector on the boundary depends on pressure, which again depends on density through the equation of state. Therefore we will need a value for density at the solid boundaries, in order to find the stresses on the solid. The second argument is about the performance of the numerical model. As explained in section 9.4 we may obtain the spacial derivatives of different field variables simply by changing the right hand side of (9.15). Thus the matrix of (9.15) needs to be factored only once. If, however, we use different domains of dependence for different field variables (velocity components and density), then the system (9.15) must be reformulated. In figure 9.11 we have indicated the two situations graphically 1) all nodes are used for the derivatives, and 2) solid boundary nodes are excluded for the derivatives of density.

As explained in section 8.2 we have considered two methods for obtaining the density at the boundaries 1) extrapolation, and 2) Neumann condition.

1) The first method works by extrapolation of the density field from the interior onto the boundary. This means that for any instant in time the density at the boundary depends only on the density in the fluid domain. The extrapolation is obtained using the MLS method (9.6) which then gives

$$\rho_i = \mathbf{e}_1 \mathbf{M}^{-1} \mathbf{b}_\rho \tag{9.56}$$

(Note that (9.6) includes $p_1(\mathbf{x}) = 1$ as a basis function and can therefore approximate the field value at any point $\mathbf{x}$ ($\mathbf{x}$ does not have to be a data point). The point $\mathbf{x}$ is in this case a point on the boundary, which then results in extrapolation of data.) It is useful to define the residual form of (9.56) in the following way

$$R_{\mathrm{e}} = \mathbf{e}_1 \mathbf{M}^{-1} \mathbf{b}_\rho - \rho_i \tag{9.57}$$

The subscript $_e$ is added to indicate that the expression is related to the *extrapolation* of density.

2) The second method for calculating density at a solid boundary is based on the Navier–Stokes equations. From these equations a Neumann condition for pressure can be established. Since pressure and density are linked through the

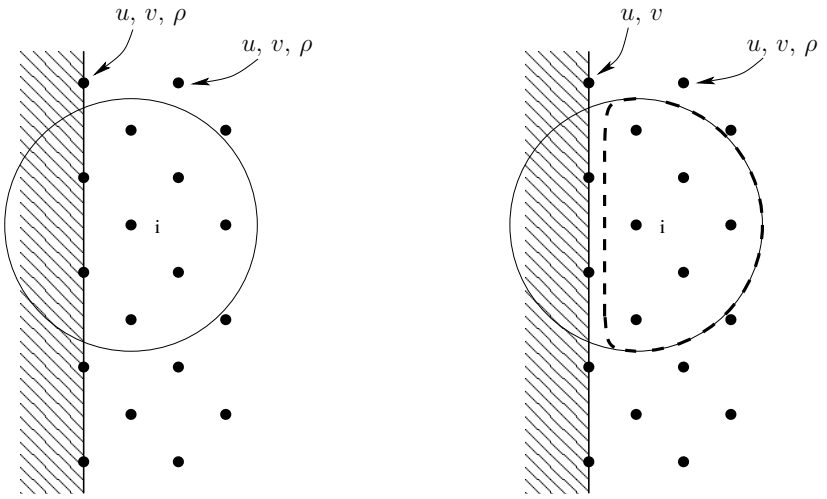

Figure 9.11: Left: Density is evaluated at the solid boundary, so a single domain of dependence for node i can be used. Right: Density is not evaluated at the solid boundary, so two different domains of dependence must be used when calculating spacial derivatives at node i.

equation of state, the condition for pressure can be translated into an equation for density. Substituting (8.5) into (8.2) gives in residual form

$$\begin{aligned} R_{\mathrm{n}} = & n_x \left[-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\partial \rho_i}{\partial x} + \mu\left(\frac{4}{3}\frac{\partial^2 u_i}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v_i}{\partial x \partial y} + \frac{\partial^2 u_i}{\partial y^2}\right)\right] \\ & + n_y \left[-\frac{\mathrm{d}p_i}{\mathrm{d}\rho}\frac{\partial \rho_i}{\partial y} + \mu\left(\frac{4}{3}\frac{\partial^2 v_i}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u_i}{\partial x \partial y} + \frac{\partial^2 v_i}{\partial x^2}\right)\right] \\ & - \rho_i \left(n_x^i a_x^i + n_y^i a_y^i\right) \end{aligned} \tag{9.58}$$

where

$$\begin{aligned} a_x^i &= \frac{\mathrm{d}u_i}{\mathrm{d}t} && x\text{-component of acceleration of node } i \\ a_y^i &= \frac{\mathrm{d}v_i}{\mathrm{d}t} && y\text{-component of acceleration of node } i \\ n_x & && x\text{-component of boundary normal} \\ n_y & && y\text{-component of boundary normal} \end{aligned} \tag{9.59}$$

The subscript $_n$ on R is meant to indicate that this residual is related to the *Neumann* condition for density. The equation for density may be associated with either a solid boundary, a free surface, or a triple point. In each case the velocity components, accelerations, and normals are given by different expressions or equations. These expressions are given in the relevant subsections.

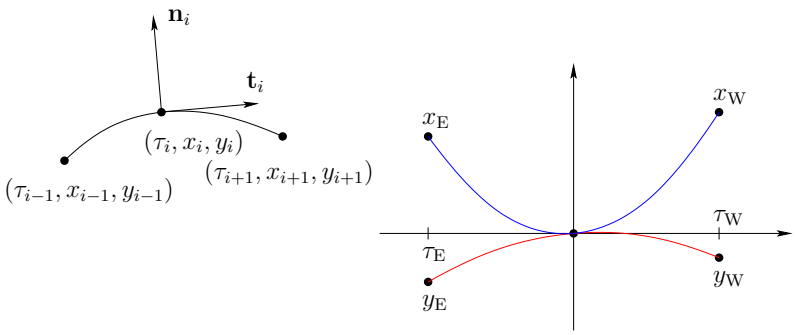

Figure 9.12: Left: Three points on a free surface. Tangent and normal are indicated. Right: Parametrization using curve length for x and y.

Free surface boundaries

The boundary conditions for a free surface are given by (8.8). Expressing these equations in residual form we get

$$\begin{aligned} U_{\mathrm{n}} &= -p_i - \frac{2}{3}\mu\left(\frac{\partial u_i}{\partial x} + \frac{\partial v_i}{\partial y}\right) + 2\mu\left(\frac{\partial u_i}{\partial x}(n_x^i)^2 + \frac{\partial v_i}{\partial y}(n_y^i)^2 + \left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right) n_x^i n_y^i\right) \\ &\quad + \kappa_i\sigma + p_{\mathrm{amb}} \\ U_{\mathrm{s}} &= 2\left(\frac{\partial u_i}{\partial x} - \frac{\partial v_i}{\partial y}\right) n_x^i n_y^i + \left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right)\left((n_y^i)^2 - (n_x^i)^2\right) \end{aligned} \tag{9.60}$$

Here the subscript $_{\mathrm{n}}$ is meant to indicate that the residual comes from the *normal* part of the stress equilibrium on the free surface. Similarly, subscript $_{\mathrm{s}}$ indicates the *shear* part.

The x-component of the normal $\mathbf{n}$ is calculated by fitting a parabola through three neighboring points, figure 9.12. The curve length is used for the parametrization, so for n_x^i we consider the points (τ_{i-1}, x_{i-1}), (τ_i, x_i), and (τ_{i+1}, x_{i+1}). The y-component of $\mathbf{n}$ is found in a similar way, so in the following we only display expressions for n_x^i. It is convenient to express the curve length directly as a function of the nodal positions in the following way

$$\begin{aligned} \tau_{\mathrm{W}} &= -\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \\ \tau_{\mathrm{P}} &= 0 \\ \tau_{\mathrm{E}} &= \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \end{aligned} \tag{9.61}$$

Note that this is a centering of the parametrization with respect to (x_i, y_i). The

first and second derivative of the parabola for n_x^i, evaluated at τ_{P} are given by

$$\begin{aligned} x_i' &= \frac{(x_{i-1}-x_i)(\tau_{\mathrm{E}})^2 - (x_{i+1}-x_i)(\tau_{\mathrm{W}})^2}{H} \\ x_i'' &= 2\frac{(x_{i+1}-x_i)(\tau_{\mathrm{W}}) - (x_{i+1}-x_i)(\tau_{\mathrm{E}})}{H} \end{aligned} \tag{9.62}$$

where the main determinant is given by

$$H = \tau_{\mathrm{W}}(\tau_{\mathrm{E}})^2 - \tau_{\mathrm{E}}(\tau_W)^2 \tag{9.63}$$

Using similar expressions in y we may compute the normal $\mathbf{n}$ and the curvature κ by

$$\begin{aligned} n_x^i &= -y_i'/L \\ n_y^i &= x_i'/L \\ \kappa_i &= \frac{x_i'y_i'' - y_i'x_i''}{L^3} \end{aligned} \tag{9.64}$$

where

$$L = \sqrt{(x_i')^2 + (y_i')^2} \tag{9.65}$$

The pressure p_i is evaluated using the equation of state and the value of ρ_i given by the density boundary condition. As already mentioned we have implemented two different methods for ρ_i given in (9.57) and (9.58).

1) If the extrapolation method (9.57) is used we could calculate pressure at the free surface as an individual step. Instead, however, we collect (9.57) and (9.60) to form a set of three equations with three unknowns. This computational procedure is slightly less efficient than solving for one field variable first (for density) and then for the velocity components. However, we have chosen this option because it enables code reuse, which simplifies the computer program.

2) If we use the Neumann condition (9.58) for density we get three equations with three unknowns. This happens because the Neumann condition couples velocity and density together in one equation. Looking at (9.58) it is seen that we need to evaluate the accelerations $\mathrm{d}u_i/\mathrm{d}t$ and $\mathrm{d}v_i/\mathrm{d}t$ at the current node i. This is done by performing the following approximation

$$\begin{aligned} \frac{\mathrm{d}u_i}{\mathrm{d}t} &= \frac{u_i - u_i^0}{\Delta t} \\ \frac{\mathrm{d}v_i}{\mathrm{d}t} &= \frac{v_i - v_i^0}{\Delta t} \end{aligned} \tag{9.66}$$

where

u_i x-component of velocity at node i at time t
v_i y-component of velocity at node i at time t
u_i^0 x-component of velocity at node i at time t^0
v_i^0 y-component of velocity at node i at time t^0
$\Delta t = t - t^0$

Here u_i^0 and v_i^0 denote previously calculated velocity components stored at time t^0. Of course t and t^0 must never be equal to each other. This means that special care is needed for the very first time step, since no previous values are available. We have chosen to solve that problem by assuming $u_i^0 = u_i^{\text{init}}$ and $v_i^0 = v_i^{\text{init}}$ for $t^0 = -\Delta t$, where $^{\text{init}}$ denotes initial values and Δt is the initial time step.

It is noted that the acceleration at the free surface nodes is actually calculated in a semi-implicit way as the time integration is carried out.

Triple points

We have tested two formulations for updating the triple points.

1) For the dynamic contact angle model we put (8.14) in residual form which gives

$$\begin{aligned} U_{\text{x}} &= u^{\text{trp}} - (u^{\text{solid}} - n_y^{\text{solid}} C_1) \\ U_{\text{y}} &= v^{\text{trp}} - (v^{\text{solid}} + n_x^{\text{solid}} C_1) \end{aligned} \tag{9.67}$$

where

$$C_1 = \frac{\cos\phi_{\text{ss}} - (n_x^{\text{solid}} n_x^{\text{fs}} + n_y^{\text{solid}} n_y^{\text{fs}})}{C_0}$$

The subscript $_{\text{x}}$ indicates that the equation comes from the x-component of the triple point velocity. Similarly, subscript $_{\text{y}}$ indicates that the equation is related to the y-component of the triple point velocity.

2) For the zero shear stress model we are able to reuse the already developed residual for the shear stress. The second part is given by (8.10) which expresses that no flow through a solid can take place. In residual form (8.10) becomes

$$U_{\text{f}} = n_x^{\text{solid}} \left(u^{\text{trp}} - u^{\text{solid}}\right) + n_y^{\text{solid}} \left(v^{\text{trp}} - v^{\text{solid}}\right) \tag{9.68}$$

We have implemented two methods for the evaluation of density at the triple points 1) extrapolation, and 2) Neumann condition. In the case of extrapolation we may reuse (9.57) directly. However, when using the Neumann condition we need to evaluate the acceleration of the triple point and the normal, see (9.58). The acceleration is approximated in the same way as for nodes on the free surface (see the previous section)

$$\begin{aligned} \frac{\mathrm{d}u^{\text{trp}}}{\mathrm{d}t} &= \frac{u^{\text{trp}} - u^0}{\Delta t} \\ \frac{\mathrm{d}v^{\text{trp}}}{\mathrm{d}t} &= \frac{v^{\text{trp}} - v^0}{\Delta t} \end{aligned} \tag{9.69}$$

where

u^{trp} x-component of velocity at time t
v^{trp} y-component of velocity at time t
u^0 x-component of velocity at time t^0
v^0 y-component of velocity at time t^0
$\Delta t = t - t^0$

The normal at a triple point is calculated by taking the average of the normals at the solid node and the free surface node, with which the triple point coincide. Thus we have

$$\begin{aligned} n_x &= \frac{n_x^{\text{fs}} + n_x^{\text{solid}}}{L} \\ n_y &= \frac{n_y^{\text{fs}} + n_y^{\text{solid}}}{L} \end{aligned} \tag{9.70}$$

where

n_x x-component of triple point normal
n_y y-component of triple point normal
n_x^{fs} x-component of normal on free surface
n_y^{fs} y-component of normal on free surface
n_x^{solid} x-component of normal on solid
n_y^{solid} y-component of normal on solid
$L = \sqrt{(n_x^{\text{fs}} + n_y^{\text{solid}})^2 + (n_y^{\text{fs}} + n_y^{\text{solid}})^2}$

9.6.2 Gradient of residual equations

In this section we develop the gradients of the various residuals defined in the previous section. These quantities are needed for the Newton-Raphson method used to calculate the unknown values on the boundary. It should be noted that we provide the gradient (partial derivatives) with respect to field values at any node j and not only with respect to the node i under consideration. For example, if the unknown of a scalar equation is u_i, then we would only need the partial derivative with respect to u_i, in order to perform Newton iterations. However, later on in section 9.7.3 we will need derivatives with respect to other variables (x_j, y_j, ...) as well. We include these expressions here, in order to keep related expressions close to each other.

Please note, that we currently have *not* implemented all gradients of some of the residuals related to triple points. This restriction will become clear in the following subsections and in sections 9.6.3 and 9.7.3.

In this section we make extensive use of the sensitivities found in section 9.4.1. Any partial derivative of the various derivatives of the field variables are taken from that section.

Gradient of $R_{\mathbf{e}}^i$ equation (9.57)

Gradient with respect to x_j

$$\frac{\partial R_{\mathrm{e}}^i}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\mathbf{e}_1 \mathbf{M}_i \frac{\partial \mathbf{b}_i^\rho}{\partial \rho_j}\right) \tag{9.71}$$

Gradient with respect to y_j

$$\frac{\partial R_{\mathrm{e}}^i}{\partial y_j} = \frac{\partial}{\partial y_j}\left(\mathbf{e}_1 \mathbf{M}_i \frac{\partial \mathbf{b}_i^\rho}{\partial \rho_j}\right) \tag{9.72}$$

Gradient with respect to u_j

$$\frac{\partial R_{\mathrm{e}}^i}{\partial u_j} = 0 \tag{9.73}$$

Gradient with respect to v_j

$$\frac{\partial R_{\mathrm{e}}^i}{\partial v_j} = 0 \tag{9.74}$$

Gradient with respect to ρ_j

$$\frac{\partial R_{\mathrm{e}}^i}{\partial \rho_j} = \frac{\partial}{\partial \rho_j}\left(\mathbf{e}_1 \mathbf{M}_i \frac{\partial \mathbf{b}_i^\rho}{\partial \rho_j}\right) - \delta_{ij}\rho_i \tag{9.75}$$

where the basis functions for the generation of $\mathbf{M}_i$ and $\mathbf{b}_i^\rho$ are given by (9.6). This means that the basis for the MLS equation includes $p_1(\mathbf{x}) = 1$.

Gradient of $R_{\mathbf{n}}^i$ equation (9.58)

Common expressions

$$\begin{aligned} F_1^i &= \frac{4}{3}\frac{\partial^2 u_i}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v_i}{\partial x \partial y} + \frac{\partial^2 u_i}{\partial y^2} \\ F_2^i &= \frac{4}{3}\frac{\partial^2 v_i}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u_i}{\partial x \partial y} + \frac{\partial^2 v_i}{\partial x^2} \\ F_3^i &= a_x^i n_x^i + a_y^i n_y^i \end{aligned} \tag{9.76}$$

The accelerations a_x^i and a_y^i as well as the normal components n_x^i and n_y^i are evaluated using the relevant expressions depending on the type of boundary under consideration, that is, solid, free surface, or triple point.

Gradient with respect to x_j

$$\begin{aligned}
\frac{\partial F_1^i}{\partial x_j} &= \frac{4}{3}\frac{\partial}{\partial x_j}\left(\frac{\partial^2 u_i}{\partial x^2}\right) + \frac{1}{3}\frac{\partial}{\partial x_j}\left(\frac{\partial^2 v_i}{\partial x \partial y}\right) + \frac{\partial}{\partial x_j}\left(\frac{\partial^2 u_i}{\partial y^2}\right) \\
\frac{\partial F_2^i}{\partial x_j} &= \frac{4}{3}\frac{\partial}{\partial x_j}\left(\frac{\partial^2 v_i}{\partial y^2}\right) + \frac{1}{3}\frac{\partial}{\partial x_j}\left(\frac{\partial^2 u_i}{\partial x \partial y}\right) + \frac{\partial}{\partial x_j}\left(\frac{\partial^2 v_i}{\partial x^2}\right) \\
\frac{\partial F_3^i}{\partial x_j} &= a_x^i \frac{\partial n_x^i}{\partial x_j} + a_y^i \frac{\partial n_y^i}{\partial x_j} \\
\frac{\partial R_{\mathrm{n}}^i}{\partial x_j} &= -\frac{\partial F_3^i}{\partial x_j}\rho_i - \frac{\mathrm{d}p_i}{\mathrm{d}\rho}\left[\frac{\partial}{\partial x_j}\left(\frac{\partial \rho_i}{\partial x}\right) n_x^i + \frac{\partial \rho_i}{\partial x}\frac{\partial n_x^i}{\partial x_j} + \frac{\partial}{\partial x_j}\left(\frac{\partial \rho_i}{\partial y}\right) n_y^i + \frac{\partial \rho_i}{\partial y}\frac{\partial n_y^i}{\partial x_j}\right] \\
&\quad + \mu\left[\frac{\partial F_1^i}{\partial x_j} n_x^i + F_1 \frac{\partial n_x^i}{\partial x_j} + \frac{\partial F_2^i}{\partial x_j} n_y^i + F_2 \frac{\partial n_y^i}{\partial x_j}\right]
\end{aligned} \tag{9.77}$$

The derivatives of the normal components n_x^i and n_y^i are evaluated depending on the type of boundary under consideration. Because solid bodies can only have prescribed kinematics we never need to reference the derivatives of n_x^i and n_y^i on solid bodies. For free surface boundaries we have placed the expressions for the derivatives of the normal components in appendix A, since the expressions are too long to be included in the main text. We have not implemented the derivatives of the normal components at a triple point. However, this could be easily done in a future work.

Gradient with respect to y_j

$$\begin{aligned}
\frac{\partial F_1^i}{\partial y_j} &= \frac{4}{3}\frac{\partial}{\partial y_j}\left(\frac{\partial^2 u_i}{\partial x^2}\right) + \frac{1}{3}\frac{\partial}{\partial y_j}\left(\frac{\partial^2 v_i}{\partial x \partial y}\right) + \frac{\partial}{\partial y_j}\left(\frac{\partial^2 u_i}{\partial y^2}\right) \\
\frac{\partial F_2^i}{\partial y_j} &= \frac{4}{3}\frac{\partial}{\partial y_j}\left(\frac{\partial^2 v_i}{\partial y^2}\right) + \frac{1}{3}\frac{\partial}{\partial y_j}\left(\frac{\partial^2 u_i}{\partial x \partial y}\right) + \frac{\partial}{\partial y_j}\left(\frac{\partial^2 v_i}{\partial x^2}\right) \\
\frac{\partial F_3^i}{\partial y_j} &= a_x^i \frac{\partial n_x^i}{\partial y_j} + a_y^i \frac{\partial n_y^i}{\partial y_j} \\
\frac{\partial R_{\mathrm{n}}^i}{\partial y_j} &= -\frac{\partial F_3^i}{\partial y_j}\rho_i - \frac{\mathrm{d}p_i}{\mathrm{d}\rho}\left[\frac{\partial}{\partial y_j}\left(\frac{\partial \rho_i}{\partial x}\right) n_x^i + \frac{\partial \rho_i}{\partial x}\frac{\partial n_x^i}{\partial y_j} + \frac{\partial}{\partial y_j}\left(\frac{\partial \rho_i}{\partial y}\right) n_y^i + \frac{\partial \rho_i}{\partial y}\frac{\partial n_y^i}{\partial y_j}\right] \\
&\quad + \mu\left[\frac{\partial F_1^i}{\partial y_j} n_x^i + F_1 \frac{\partial n_x^i}{\partial y_j} + \frac{\partial F_2^i}{\partial y_j} n_y^i + F_2 \frac{\partial n_y^i}{\partial y_j}\right]
\end{aligned} \tag{9.78}$$

Gradient with respect to u_j

$$\begin{aligned}
\frac{\partial F_1^i}{\partial u_j} &= \frac{4}{3}\frac{\partial}{\partial u_j}\left(\frac{\partial^2 u_i}{\partial x^2}\right) + \frac{\partial}{\partial u_j}\left(\frac{\partial^2 u_i}{\partial y^2}\right) \\
\frac{\partial F_2^i}{\partial u_j} &= \frac{1}{3}\frac{\partial}{\partial u_j}\left(\frac{\partial^2 u_i}{\partial x \partial y}\right) \\
\frac{\partial F_3^i}{\partial u_j} &= \begin{cases} \delta_{ij}\dfrac{n_x^i}{t-t_0} & \text{if node } i \text{ is on a free surface or triple point} \\ 0 & \text{if node } i \text{ is on a solid} \end{cases} \\
\frac{\partial R_{\mathrm{n}}^i}{\partial u_j} &= -\rho_i \frac{\partial F_3^i}{\partial u_j} + \mu\left(\frac{\partial F_1^i}{\partial u_j} n_x^i + \frac{\partial F_2^i}{\partial u_j} n_y^i\right)
\end{aligned} \tag{9.79}$$

The branching on $\partial F_3^i / \partial u_j$ is due to the fact that solids in this work has always prescribed kinematics, and therefore only depend on time t.

Gradient with respect to v_j

$$\begin{aligned}
\frac{\partial F_1^i}{\partial v_j} &= \frac{1}{3}\frac{\partial}{\partial v_j}\left(\frac{\partial^2 v_i}{\partial x \partial y}\right) \\
\frac{\partial F_2^i}{\partial v_j} &= \frac{4}{3}\frac{\partial}{\partial v_j}\left(\frac{\partial^2 v_i}{\partial y^2}\right) + \frac{\partial}{\partial v_j}\left(\frac{\partial^2 v_i}{\partial x^2}\right) \\
\frac{\partial F_3^i}{\partial v_j} &= \begin{cases} \delta_{ij}\dfrac{n_y^i}{t-t_0} & \text{if node } i \text{ is on a free surface or triple point} \\ 0 & \text{if node } i \text{ is on a solid} \end{cases} \\
\frac{\partial R_{\mathrm{n}}^i}{\partial v_j} &= -\rho_i \frac{\partial F_3^i}{\partial v_j} + \mu\left(\frac{\partial F_1^i}{\partial v_j} n_x^i + \frac{\partial F_2^i}{\partial v_j} n_y^i\right)
\end{aligned} \tag{9.80}$$

Gradient with respect to ρ_j

$$\begin{aligned}
\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_j} = -\delta_{ij} &\left(F_3^i + \frac{\mathrm{d}^2 p_i}{\mathrm{d}\rho^2}\left[\frac{\partial \rho_i}{\partial x} n_x^i + \frac{\partial \rho_i}{\partial y} n_y^i\right]\right) \\
&- \frac{\mathrm{d}p_i}{\mathrm{d}\rho}\left[\frac{\partial}{\partial \rho_j}\left(\frac{\partial \rho_i}{\partial x}\right) + \frac{\partial}{\partial \rho_j}\left(\frac{\partial \rho_i}{\partial y}\right)\right]
\end{aligned} \tag{9.81}$$

Gradient of U^i_{n} equation (9.60)

Gradient with respect to x_j or y_j

$$\begin{aligned}\frac{\partial U^i_{\mathrm{n}}}{\partial(\cdot)} = &-\frac{2}{3}\mu\left[\frac{\partial}{\partial(\cdot)}\left(\frac{\partial u_i}{\partial x}\right)+\frac{\partial}{\partial(\cdot)}\left(\frac{\partial v_i}{\partial y}\right)\right]\\ &+2\mu\left[\frac{\partial}{\partial(\cdot)}\left(\frac{\partial u_i}{\partial x}\right)(n^i_x)^2+\frac{\partial}{\partial(\cdot)}\left(\frac{\partial v_i}{\partial y}\right)(n^i_y)^2+2\left(\frac{\partial u_i}{\partial x}\frac{\partial n^i_x}{\partial(\cdot)}n^i_x+\frac{\partial v_i}{\partial y}\frac{\partial n^i_y}{\partial(\cdot)}n^i_y\right)\right.\\ &\left.+\left(\frac{\partial}{\partial(\cdot)}\left(\frac{\partial u_i}{\partial y}\right)+\frac{\partial}{\partial(\cdot)}\left(\frac{\partial v_i}{\partial x}\right)\right)n^i_x n^i_y+\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\left(\frac{\partial n^i_x}{\partial(\cdot)}n^i_y+\frac{\partial n^i_y}{\partial(\cdot)}n^i_x\right)\right]\\ &+\sigma\frac{\partial\kappa}{\partial(\cdot)}\end{aligned} \tag{9.82}$$

where $(\cdot)$ denotes either of x_j or y_j. Here σ is the surface tension as in (8.8). The derivatives of the normal components n^i_x and n^i_y and curvature κ are given in appendix A. Note that the derivatives of the normal components and curvature are zero, unless node i and node j are neighboring (or identical) nodes on a free surface.

Gradient with respect to u_j

$$\frac{\partial U^i_{\mathrm{n}}}{\partial u_j} = -\frac{2}{3}\mu\left[\frac{\partial}{\partial u_j}\left(\frac{\partial u_i}{\partial x}\right)\right]+2\mu\left[\frac{\partial}{\partial u_j}\left(\frac{\partial u_i}{\partial x}\right)(n^i_x)^2+\frac{\partial}{\partial u_j}\left(\frac{\partial u_i}{\partial y}\right)n^i_x n^i_y\right] \tag{9.83}$$

Gradient with respect to v_j

$$\frac{\partial U^i_{\mathrm{n}}}{\partial v_j} = -\frac{2}{3}\mu\left[\frac{\partial}{\partial v_j}\left(\frac{\partial v_i}{\partial y}\right)\right]+2\mu\left[\frac{\partial}{\partial v_j}\left(\frac{\partial v_i}{\partial y}\right)(n^i_y)^2+\frac{\partial}{\partial v_j}\left(\frac{\partial v_i}{\partial x}\right)n^i_x n^i_y\right] \tag{9.84}$$

Gradient with respect to ρ_j

$$\frac{\partial U^i_{\mathrm{n}}}{\partial \rho_j} = -\delta_{ij}\frac{\mathrm{d}p_i}{\mathrm{d}\rho} \tag{9.85}$$

Gradient of U^i_{s} equation (9.60)

Gradient with respect to x_j or y_j

$$\begin{aligned}\frac{\partial U^i_{\mathrm{s}}}{\partial(\cdot)} = &\,2\left[\frac{\partial}{\partial(\cdot)}\left(\frac{\partial u_i}{\partial x}\right)-\frac{\partial}{\partial(\cdot)}\left(\frac{\partial v_i}{\partial y}\right)\right]n^i_x n^i_y\\ &+2\left(\frac{\partial u_i}{\partial x}-\frac{\partial v_i}{\partial y}\right)\left(\frac{\partial n^i_x}{\partial(\cdot)}n^i_y+\frac{\partial n^i_y}{\partial(\cdot)}n^i_x\right)\\ &+\left[\frac{\partial}{\partial(\cdot)}\left(\frac{\partial u_i}{\partial y}\right)+\frac{\partial}{\partial(\cdot)}\left(\frac{\partial v_i}{\partial x}\right)\right]\left((n^i_y)^2-(n^i_x)^2\right)\\ &+2\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\left(\frac{\partial n^i_y}{\partial(\cdot)}n^i_y-\frac{\partial n^i_x}{\partial(\cdot)}n^i_x\right)\end{aligned} \tag{9.86}$$

where $(\cdot)$ denotes either of x_j or y_j.

Gradient with respect to u_j

$$\frac{\partial U_{\mathrm{s}}^i}{\partial u_j} = 2\left[\frac{\partial}{\partial u_j}\left(\frac{\partial u_i}{\partial x}\right)\right] n_x^i n_y^i + \left[\frac{\partial}{\partial u_j}\left(\frac{\partial u_i}{\partial y}\right)\right]\left((n_y^i)^2 - (n_x^i)^2\right) \tag{9.87}$$

Gradient with respect to v_j

$$\frac{\partial U_{\mathrm{s}}^i}{\partial v_j} = 2\left[-\frac{\partial}{\partial v_j}\left(\frac{\partial v_i}{\partial y}\right)\right] n_x^i n_y^i + \left[\frac{\partial}{\partial v_j}\left(\frac{\partial v_i}{\partial x}\right)\right]\left((n_y^i)^2 - (n_x^i)^2\right) \tag{9.88}$$

Gradient with respect to ρ_j

$$\frac{\partial U_{\mathrm{s}}^i}{\partial \rho_j} = 0 \tag{9.89}$$

Gradient of U_{x}^i equation (9.67)

Below we provide the gradients of U_{x}^i with respect to u_j, v_j, and ρ_j. These gradients are sufficient for solving the residual equations at a triple point, in order to find the unknown values u_i, v_i, and ρ_i. However, gradients with respect x_j and y_j are currently not implemented. This restriction means that triple points are currently not supported for implicit time integration, as will be discussed in section 9.7.3.

Note that solids in this work always have prescribed kinematics – therefore derivatives with respect to u_j or v_j are never referenced, when node j belongs to a solid. This fact has been taken into account in the following.

Gradient with respect to u_j

$$\frac{\partial U_{\mathrm{x}}^i}{\partial u_j} = \delta_{ij} \tag{9.90}$$

Gradient with respect to v_j

$$\frac{\partial U_{\mathrm{x}}^i}{\partial v_j} = 0 \tag{9.91}$$

Gradient with respect to ρ_j

$$\frac{\partial U_{\mathrm{x}}^i}{\partial \rho_j} = 0 \tag{9.92}$$

Gradient of U_{y}^i equation (9.67)

The same comment as in the previous subsection 9.6.2 applies to this subsection.

Gradient with respect to u_j

$$\frac{\partial U_{\mathrm{y}}^i}{\partial u_j} = 0 \tag{9.93}$$

Gradient with respect to v_j

$$\frac{\partial U_{\mathrm{y}}^i}{\partial v_j} = \delta_{ij} \tag{9.94}$$

Gradient with respect to ρ_j

$$\frac{\partial U_{\mathrm{y}}^i}{\partial \rho_j} = 0 \tag{9.95}$$

Gradient of U_{f}^i equation (9.68)

The same comment as in section 9.6.2 applies to this subsection.

Gradient with respect to u_j

$$\frac{\partial U_{\mathrm{f}}^i}{\partial u_j} = \delta_{ij} n_x^{\mathrm{solid}} \tag{9.96}$$

Gradient with respect to v_j

$$\frac{\partial U_{\mathrm{f}}^i}{\partial v_j} = \delta_{ij} n_y^{\mathrm{solid}} \tag{9.97}$$

Gradient with respect to ρ_j

$$\frac{\partial U_{\mathrm{f}}^i}{\partial \rho_j} = 0 \tag{9.98}$$

9.6.3 Enforcing boundary conditions

In the previous sections 9.6.1 and 9.6.2 we developed the equations and expressions needed in order to calculate the unknown values on the boundary. There is, however, one detail left to consider, and that is how many dependencies between the nodes on the boundaries should be maintained?

We have considered two different options, which we denote by *explicit* and *implicit* boundary conditions. These terms reflect that in the explicit method we calculate boundary values for each node one by one, while in the implicit method we solve a set of equations for the boundary values in all nodes simultaneously. We have indicated the two methods in figure 9.13.

1) The explicit method is very useful because the unknown value on the boundary only couples with interior nodes, where the value of the field variables are

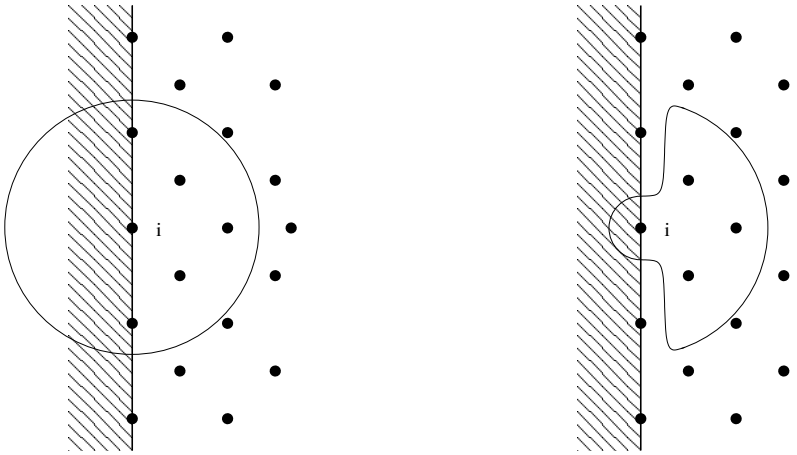

Figure 9.13: Left: *Implicit method*: Boundary node couples with all nodes within smoothing length. Right: *Explicit method*: Boundary node couples with nodes from the interior of the fluid domain only.

constant (at every instant of time). This means that the values of e.g. density on a solid boundary can be calculated at one node at a time and in any order.

2) The implicit method is the result of including all nodes within the smoothing length of every node on the boundary. Clearly, a specific node on the boundary will in this case depend on other boundary nodes, where the value of some of the field variables are not yet known. This results in a set of equations that must be solved simultaneously for all unknowns.

An advantage of the explicit method is that it is efficient and simple to implement. The main drawback is that the smoothing length of the nodes on the boundary must be rather large in order to include enough interior nodes to make (9.15) well-posed. This is especially true for nodes at e.g. a 90 degree corner which for a given smoothing length and regular layout of nodes has fewer neighbors than for example an interior node with the same smoothing length. On the other hand the implicit method includes all neighbors and can therefore do with just a slight increase of the smoothing length compared to interior nodes. However, the implicit method requires the solution of a set of equations, which is more demanding with respect to computational effort. The method is also less straight forward to implement due to the assembly of the system of equations.

In the following two subsections we give details on how the boundary values are computed using either of the two methods.

Explicit boundary conditions

If we neglect the coupling between nodes on the boundary, then we can calculate the unknown boundary values at one node at a time. Thus for a solid boundary we solve

$$R^i_{\rm n}(\rho_i) = 0 \quad \text{or} \quad R^i_{\rm e}(\rho_i) = 0 \tag{9.99}$$

defined at node i for the unknown density ρ_i.

At a free surface we solve the following set of equations

$$\begin{aligned} R^i_{\rm n}(u_i, v_i, \rho_i) = 0 \quad &\text{or} \quad R^i_{\rm e}(\rho_i) = 0 \\ U^i_{\rm n}(u_i, v_i, \rho_i) = 0 \quad & \\ V^i_{\rm s}(u_i, v_i) = 0 \quad & \end{aligned} \tag{9.100}$$

defined at node i for the three unknowns u_i, v_i, and ρ_i.

At a triple point we solve the following set of equations

$$\begin{aligned} R^i_{\rm n}(u_i, v_i, \rho_i) = 0 \quad &\text{or} \quad R^i_{\rm e}(\rho_i) = 0 \\ \left.\begin{aligned} U^i_{\rm x}(u_i, v_i) = 0 \\ U^i_{\rm y}(u_i, v_i) = 0 \end{aligned}\right\} \quad &\text{or} \quad \left\{\begin{aligned} U^i_{\rm f}(u_i, v_i) = 0 \\ U^i_{\rm s}(u_i, v_i) = 0 \end{aligned}\right. \end{aligned} \tag{9.101}$$

defined at node i for the three unknowns u_i, v_i, and ρ_i.

The boundary values are found one by one, using two nested loops. The outer loop loops over solid boundaries, free surfaces, and then triple points. The inner loop loops over the nodes belonging to a boundary object (a solid, free surface, or triple point).

Implicit boundary conditions

If we take into account the coupling between nodes on the boundary, then a set of simultaneous equations develop. The vector of unknowns is given by

$$\mathbf{z} = (\mathbf{solid}_1, \ldots, \mathbf{freesurface}_1, \ldots, \mathbf{triplepoint}_1, \ldots)^{\rm T} \tag{9.102}$$

(ellipsis indicate that e.g. more than one solid object might exist) where

$$\begin{aligned} \mathbf{solid}_1 &= (\rho_1, \rho_2, \ldots)^{\rm T} \\ &\vdots \\ \mathbf{freesurface}_1 &= (u_1, v_1, \rho_1, u_2, v_2, \rho_2, \ldots)^{\rm T} \\ &\vdots \\ \mathbf{triplepoint}_1 &= (u, v, \rho)^{\rm T} \\ &\vdots \end{aligned} \tag{9.103}$$

Note that this notation uses a local numbering of the field variables for each of the boundary objects (ρ_1 belonging to $\mathbf{solid}_1$ is not the same as ρ_1 belonging to $\mathbf{freesurface}_1$, etc). When assembling the complete system of equations a global numbering is used.

The residuals are collected in a right hand side vector $\mathbf{r}$, and a matrix of coefficients $\mathbf{A}$ is obtained from the gradient information from section 9.6.2

$$\mathbf{Az} = \mathbf{r} \tag{9.104}$$

This equation is nonlinear if the Neumann condition for density is used in combination with a nonlinear equation of state, otherwise it is linear. In the nonlinear case it is necessary to iterate on the equation above.

9.6.4 Sensitivity analysis of explicit boundary conditions

In this section we provide the sensitivity analysis of the *explicit* boundary conditions of section 9.6.3. We have not implemented the sensitivities related to the *implicit* enforcement of the boundary conditions.

The sensitivity of the boundary conditions express how (i.e. at what rate) the value on the boundary changes due to a perturbation of one of the variables on which it depends. In the following we develop the expressions for 1) density on solid boundaries, and 2) velocity components and density on free surfaces. This information is needed if an implicit time integration method is to be used (section 9.7.3).

We have not implemented the sensitivities related to triple points in the present version of the simulation program (the sensitivities are, however, derived in the same manner as those for e.g. a free surface node).

Sensitivities at solid boundaries

The starting point for the sensitivity analysis is the residual equation for density, either (9.57) or (9.58). Beginning with the extrapolation method for density we may write

$$\begin{aligned} R^i_{\mathrm{e}}\big(&x_{1,\ldots,N},\\ &y_{1,\ldots,N},\\ &\rho_{1,\ldots,i-1},\\ &\rho_i(x_{1,\ldots,N}, y_{1,\ldots,N}, \rho_{1,\ldots,i-1,i+1,\ldots,N}),\\ &\rho_{i+1,\ldots,N}\big) = 0 \end{aligned} \tag{9.105}$$

where N is the number of nodes within smoothing length, excluding neighboring nodes on the solid boundary (see figure 9.13). The notation indicates that ρ_i is an implicit function of the location (x, y) of any node within smoothing length, and also the density ρ of any node within smoothing length – *except* ρ_i *itself*. On

the other hand the residual function R_{e}^i depends on x, y, and ρ of *all* variables within the smoothing length. Taking the total derivative of (9.105) with respect to x_j gives

$$\frac{\mathrm{d}R_{\mathrm{e}}^i}{\mathrm{d}x_j} = \frac{\partial R_{\mathrm{n}}^i}{\partial x_j} + \frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\frac{\partial \rho_i}{\partial x_j} = 0 \tag{9.106}$$

Note that is equation holds only if (9.105) is satisfied. Rearranging gives

$$\frac{\partial \rho_i}{\partial x_j} = -\left(\frac{\partial R_{\mathrm{e}}^i}{\partial \rho_i}\right)^{-1}\frac{\partial R_{\mathrm{n}}^i}{\partial x_j} \tag{9.107}$$

which is the sensitivity of the density at node i on the solid boundary, with respect to x at node j, $j = 1, \ldots, N$. Similarly, we may obtain these expressions for the sensitivities with respect y and ρ, respectively

$$\frac{\partial \rho_i}{\partial y_j} = -\left(\frac{\partial R_{\mathrm{e}}^i}{\partial \rho_i}\right)^{-1}\frac{\partial R_{\mathrm{n}}^i}{\partial y_j} \tag{9.108}$$

valid for $j = 1, \ldots, N$ and

$$\frac{\partial \rho_i}{\partial \rho_j} = -\left(\frac{\partial R_{\mathrm{e}}^i}{\partial \rho_i}\right)^{-1}\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_j} \tag{9.109}$$

valid for $j = 1, \ldots, i-1, i+1, \ldots, N$. Note the common factor on the right hand side of the expressions for the sensitivities.[1]

In the case of the Neumann condition for density we may write the residual equation in the following way, showing the dependencies

$$\begin{aligned} R_{\mathrm{n}}^i\big(&x_{1,\ldots,N},\\ &y_{1,\ldots,N},\\ &u_{1,\ldots,N},\\ &v_{1,\ldots,N},\\ &\rho_{1,\ldots,i-1},\\ &\rho_i(x_{1,\ldots,N}, y_{1,\ldots,N}, u_{1,\ldots,N}, v_{1,\ldots,N}, \rho_{1,\ldots,i-1,i+1,N}),\\ &\rho_{i+1,N}\big) = 0 \end{aligned} \tag{9.110}$$

[1]This kind of sensitivity analysis is known as the *direct* method. For each sensitivity one right hand side is generated, which is then "divided" by a common "factor" (in general the "factor" is a matrix, which is decomposed once). The *adjoint* method is in many cases much more efficient, since only a single right hand side develops. Although the adjoint method is applicable for this problem – with benefit – we have not used it. This decision was made because of increased memory requirements and extra data structures needed for the adjoint method.

Assuming that (9.110) is satisfied and using the same procedure as before we get the sensitivities

$$\frac{\partial \rho_i}{\partial x_j} = -\left(\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\right)^{-1} \frac{\partial R_{\mathrm{n}}^i}{\partial x_j} \tag{9.111}$$

valid for $j = 1, \ldots, N$, and

$$\frac{\partial \rho_i}{\partial y_j} = -\left(\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\right)^{-1} \frac{\partial R_{\mathrm{n}}^i}{\partial y_j} \tag{9.112}$$

valid for $j = 1, \ldots, N$, and

$$\frac{\partial \rho_i}{\partial u_j} = -\left(\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\right)^{-1} \frac{\partial R_{\mathrm{n}}^i}{\partial u_j} \tag{9.113}$$

valid for $j = 1, \ldots, N$, and

$$\frac{\partial \rho_i}{\partial v_j} = -\left(\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\right)^{-1} \frac{\partial R_{\mathrm{n}}^i}{\partial v_j} \tag{9.114}$$

valid for $j = 1, \ldots, N$, and

$$\frac{\partial \rho_i}{\partial \rho_j} = -\left(\frac{\partial R_{\mathrm{n}}^i}{\partial \rho_i}\right)^{-1} \frac{\partial R_{\mathrm{n}}^i}{\partial \rho_j} \tag{9.115}$$

valid for $j = 1, \ldots, i-1, i+1, \ldots, N$. Note the common factor on the right side. The partial derivatives needed for the sensitivities are taken from section 9.6.2.

Sensitivities at free surfaces

The sensitivity on free surfaces is found in the same way as for the case with a solid boundary. The only difference is that we now have multiple dependent values on the boundary. Since these values are coupled to each other a system of equations develops for the sensitivities. In the most general case we have

$$\begin{aligned}
&F^i\big(x_{1,\ldots,N},\\
&\quad y_{1,\ldots,N},\\
&\quad u_{1,\ldots,i-1},\\
&\quad u_i(x_{1,\ldots,N}, y_{1,\ldots,N}, u_{1,\ldots,i-1}, u_{i+1,\ldots,N}, v_{1,\ldots,N}, \rho_{1,\ldots,N}),\\
&\quad u_{i+1,\ldots,N},\\
&\quad v_{1,\ldots,i-1},\\
&\quad v_i(x_{1,\ldots,N}, y_{1,\ldots,N}, u_{1,\ldots,N}, v_{1,\ldots,i-1}, v_{i+1,\ldots,N}, \rho_{1,\ldots,N}),\\
&\quad v_{i+1,\ldots,N},\\
&\quad \rho_{1,\ldots,i-1},\\
&\quad \rho_i(x_{1,\ldots,N}, y_{1,\ldots,N}, u_{1,\ldots,N}, v_{1,\ldots,N}, \rho_{1,\ldots,i-1}, \rho_{i+1,\ldots,N}),\\
&\quad \rho_{i+1,\ldots,N}\big) = 0
\end{aligned} \tag{9.116}$$

where F^i denotes either of $R^i_{\rm e}$, $R^i_{\rm n}$, $U^i_{\rm n}$, or $U^i_{\rm s}$. Taking the total derivative and rearranging in the same fashion as in the previous section gives

$$\begin{pmatrix} \dfrac{\partial u^i}{\partial x_j} \\ \dfrac{\partial v^i}{\partial x_j} \\ \dfrac{\partial \rho^i}{\partial x_j} \end{pmatrix} = - \begin{bmatrix} \dfrac{\partial F^i_1}{\partial u^i} & \dfrac{\partial F^i_1}{\partial v^i} & \dfrac{\partial F^i_1}{\partial \rho^i} \\ \dfrac{\partial F^i_2}{\partial u^i} & \dfrac{\partial F^i_2}{\partial v^i} & \dfrac{\partial F^i_2}{\partial \rho^i} \\ \dfrac{\partial F^i_3}{\partial u^i} & \dfrac{\partial F^i_3}{\partial v^i} & \dfrac{\partial F^i_3}{\partial \rho^i} \end{bmatrix}^{-1} \begin{pmatrix} \dfrac{\partial F^i_1}{\partial x_j} \\ \dfrac{\partial F^i_2}{\partial x_j} \\ \dfrac{\partial F^i_3}{\partial x_j} \end{pmatrix} \tag{9.117}$$

where

$$\begin{aligned} F^i_1 &= R^i_{\rm e} \quad \text{or} \quad F^i_1 = R^i_{\rm n} \\ F^i_2 &= U^i_{\rm n} \\ F^i_3 &= U^i_{\rm s} \end{aligned} \tag{9.118}$$

depending on what scheme for density is used.

Similar equations can be derived for the sensitivities with respect to y_j, u_j, v_j, and ρ_j. It is noted that the matrix of coefficients is the same whether the sensitivity is with respect to x, y, u, v, or ρ. So only one factorization is needed. Again the partial derivatives that appear in these expressions are taken from from section 9.6.2. Note that the sensitivity analysis is based on the satisfaction of (9.116). Thus, we have to solve this equation first (which gives the unknown values on the boundary) and then use the current values when evaluating the partial derivatives.

9.7 Time integration

In section 9.5 we have seen how the spacial discretization of the Navier–Stokes equations is carried out. In section 9.6 we have also seen how the boundary conditions are implemented. With these things settled we are ready to carry on with the time integration method. We have examined two types of methods, namely explicit and implicit Runge–Kutta schemes. In the next section we first present the organization of the field variables into a *state vector*, then we present the explicit time integrator, which is followed by a section on implicit time integration.

9.7.1 The state vector

The Runge–Kutta methods apply to systems of ordinary differential equations. Each differential equation should be an initial value problem. At first it might seem that we cannot use this class of methods for the Navier–Stokes equations, since they are partial differential equations. However, we have employed the semi-discretization technique, which transforms partial differential equations into a set

of ordinary differential equations. In fact each interior node in the computational domain will be advanced in time as if it were governed by a scalar ordinary differential equation. Another way of putting it, is that the Navier–Stokes equations must be satisfied at each node of the interior of the fluid domain

$$\begin{aligned}
\frac{\mathrm{d}x_i}{\mathrm{d}t} &= u_i \\
\frac{\mathrm{d}y_i}{\mathrm{d}t} &= v_i \\
\frac{\mathrm{d}u_i}{\mathrm{d}t} &= f_i^u(x_1, x_2, \ldots x_N, y_1, \ldots, y_N, u_1, \ldots u_N, v_1, \ldots, v_n, \rho_1, \ldots, \rho_N) \\
\frac{\mathrm{d}v_i}{\mathrm{d}t} &= f_i^v(x_1, x_2, \ldots x_N, y_1, \ldots, y_N, u_1, \ldots u_N, v_1, \ldots, v_n, \rho_1, \ldots, \rho_N) \\
\frac{\mathrm{d}\rho_i}{\mathrm{d}t} &= f_i^\rho(x_1, x_2, \ldots x_N, y_1, \ldots, y_N, u_1, \ldots u_N, v_1, \ldots, v_n, \rho_1, \ldots, \rho_N)
\end{aligned} \tag{9.119}$$

for i looping over all interior nodes. The right hand sides f^u, f^v, f^ρ are aliases for the right hand side of the momentum equations (9.47) and the continuity equation (9.48). In the notation above we have included *all* possible dependencies for node i (N is the total number of nodes, including interior nodes as well as nodes on the boundary). In practice only nodes within the smoothing length of node i contribute to the right hand side. The important thing which we wanted to stress is that the interior nodes may couple with nodes on the boundary.

The discrete variables may be divided into two groups 1) variables advanced by the time integration method, and 2) variables governed by the boundary conditions. The variables belonging to the first group are denoted by the *state variables*. These values are stored in a vector $\mathbf{s}$ defined in the following way

$$\mathbf{s} = (\mathbf{fluid}, \mathbf{freesurface}_1, \ldots, \mathbf{triplepoint}_1, \ldots)^{\mathrm{T}} \tag{9.120}$$

where

$$\begin{aligned}
\mathbf{fluid} &= (x_1, y_1, u_1, v_1, \rho_1, \ldots, x_N, y_N, u_N, v_N, \rho_N)^{\mathrm{T}} \\
\mathbf{freesurface}_1 &= (x_1, y_1, \ldots, x_N, y_N)^{\mathrm{T}} \\
&\vdots \\
\mathbf{triplepoint}_1 &= (x, y)^{\mathrm{T}} \\
&\vdots
\end{aligned} \tag{9.121}$$

Note that N in each instance denotes the number of nodes belonging to the object with which it is associated (the number of nodes N in the fluid domain is not the same variable as N for a free surface, etc). As indicated, the simulation program supports only the definition of a single fluid domain. On the other hand any number of solids, free surfaces and triple points is allowed. It should be noted

that the first and last node of a free surface is excluded from the state vector, if the free surface is ended by triple points.

The values governed by the boundary conditions are summarized here

$$\begin{aligned} x,\, y,\, u,\, v, \text{ and } \rho &\quad \text{for all nodes on all solid boundaries} \\ u,\, v, \text{ and } \rho &\quad \text{for all nodes on all free surfaces and triple points} \end{aligned} \tag{9.122}$$

If we take the union of the variables in (9.120) and (9.122) we see that all variables are accounted for.

We may write the system of equations for time integration in the following generic form

$$\dot{\mathbf{s}} = \mathbf{f}(t, \mathbf{s}) \tag{9.123}$$

where $\mathbf{s}$ is given by (9.120) and the right hand side function $\mathbf{f}$ is composed in the following way

$$\mathbf{f} = (\mathbf{f}_{\text{fluid}}, \mathbf{f}_{\text{freesurface}_1}, \ldots, \mathbf{f}_{\text{triplepoint}_1}, \ldots)^{\mathrm{T}} \tag{9.124}$$

where

$$\begin{aligned} \mathbf{f}_{\text{fluid}} &= (u_1, v_1, \mathrm{d}u_1/\mathrm{d}t, \mathrm{d}v_1/\mathrm{d}t, \mathrm{d}\rho_1/\mathrm{d}t, \ldots, \\ &\qquad u_N, v_N, \mathrm{d}u_N/\mathrm{d}t, \mathrm{d}v_N/\mathrm{d}t, \mathrm{d}\rho_N/\mathrm{d}t)^{\mathrm{T}} \\ \mathbf{f}_{\text{freesurface}_1} &= (u_1, v_1, \ldots, u_N, v_N)^{\mathrm{T}} \\ &\vdots \\ \mathbf{f}_{\text{triplepoint}_1} &= (u, v)^{\mathrm{T}} \\ &\vdots \end{aligned} \tag{9.125}$$

9.7.2 Explicit time integration

We have implemented an explicit Runge–Kutta scheme, known as the *Nystrom* scheme, which is quoted in Iserles [47]. The scheme has three stages given by

$$\begin{aligned} \mathbf{s}_1 &= \mathbf{s}^k \\ \mathbf{s}_2 &= \mathbf{s}^k + h\left(A_{21}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1)\right) \\ \mathbf{s}_3 &= \mathbf{s}^k + h\left(A_{31}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + A_{32}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2)\right) \end{aligned} \tag{9.126}$$

where matrix $\mathbf{A}$ contains the following coefficients

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 2/3 & 0 & 0 \\ 0 & 2/3 & 0 \end{bmatrix} \tag{9.127}$$

and vector $\mathbf{c}$ holds these fractions of the time step

$$\mathbf{c} = (\ 0 \quad 2/3 \quad 2/3\ )^{\mathrm{T}} \tag{9.128}$$

and h is the size of the time step. The Nystrom scheme has a third order solution and an embedded second order solution, which are given in the following way, respectively.

Third order solution

$$\mathbf{s}^{k+1} = \mathbf{s}^k + h\left(\frac{1}{4}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + \frac{2}{3}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2) + \frac{2}{3}\mathbf{f}(t^k + c_3 h, \mathbf{s}_3)\right) \tag{9.129}$$

Second order solution

$$\tilde{\mathbf{s}}^{k+1} = \mathbf{s}^k + h\left(\frac{1}{4}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + \frac{3}{4}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2)\right) \tag{9.130}$$

The truncation error can be expressed in the following way[2]

$$\begin{aligned} ||\mathbf{E}_4|| &= ||\mathbf{s}^{k+1} - \mathbf{s}(t_{k+1})|| = O(h^4) \\ ||\mathbf{E}_3|| &= ||\tilde{\mathbf{s}}^{k+1} - \mathbf{s}(t_{k+1})|| = O(h^3) = C_1 h^3 + O(h^4) \end{aligned} \tag{9.131}$$

where $\mathbf{s}(t_{k+1})$ denotes the exact (unknown) solution and C_1 is an unknown constant. The error can be *estimated* by subtracting the third order and second order solution from each order which gives

$$||\mathbf{e}|| = ||\tilde{\mathbf{s}}^{k+1} - \mathbf{s}^{k+1}|| = C_1 h^3 + O(h^4) \tag{9.132}$$

The presence of the term $O(h^4)$ in the formula above indicate that the error is only estimated. Based on the error estimate we can develop a variable step size strategy. The starting point is to consider two steps of different size, which would give two different estimations of the error

$$\begin{aligned} ||\mathbf{e}_1|| &= C_1 h_1^3 + O(h^4) \\ ||\mathbf{e}_2|| &= C_2 h_2^3 + O(h^4) \end{aligned} \tag{9.133}$$

Now if $h \to 0$ then the higher order terms $O(h^4)$ go to zero at a faster rate than the terms with h^3. In the limiting case this means that $C_1 - C_2 \to 0$ for $h \to 0$, in which case we may write

$$\frac{||\mathbf{e}_1||}{||\mathbf{e}_2||} = \left(\frac{h_1}{h_2}\right)^3 \tag{9.134}$$

[2]Note that a time integrator is said to have order p, if the truncation error is of order $p+1$. This is a different terminology as compared with, e.g. finite difference schemes which are said to be of order q is the truncation error is of order q.

This expression shows how the (estimated) error scales with the size of the time step. Thus if we substitute $||\mathbf{e}_1||$ with the allowed *tolerance* ε, we may compute the corresponding step size by

$$h_{\text{new}} = r h_{\text{cur}} \left(\frac{\varepsilon}{||\mathbf{e}||} \right)^{1/3} \tag{9.135}$$

where

h_{new}	step size for the next time step
h_{cur}	the current step size
$||\mathbf{e}||$	2-norm of the error vector $(\mathbf{s}^{k+1} - \tilde{\mathbf{s}}^{k+1})$ from the current time step
ε	allowed error
r	safety factor

The safety factor is set to $r = 0.95$. The allowed error should reflect the magnitude of the entries in $\mathbf{s}$, that is, typically a relative measure is desired. A simple way of combining a relative tolerance and an absolute tolerance is to define the allowed error in the following way

$$\varepsilon = \varepsilon_r ||\mathbf{s}^{k+1}|| + \varepsilon_a \tag{9.136}$$

where

ε_r	relative tolerance
ε_a	absolute tolerance

We have implemented the variable time step strategy as given by (9.135) and (9.136) in the simulation program. If the allowed error is exceeded the step is recalculated, using the newly estimated time step. Alternatively, the simulation program can be used with a fixed size of the time step.

It should be noted that the error estimate (9.132) is actually only valid, if we use the second order solution $\tilde{\mathbf{s}}^{k+1}$ to advance the system. However, it is tempting to use the third order solution $\mathbf{s}^{k+1}$ instead, since it is more accurate, although this practice is not strictly correct. We have chosen to use this "trick", because we in general will use a fixed time step anyhow (the tilde on the second order solution indicates that it is used for error estimation).

Time stepping procedure

Having defined the state variables of the time integration problem, we are now ready to establish the procedure for the time integration. The starting point is a configuration where all state variables $\mathbf{s}_0$ are known at time $t = t_0$. Using the kinematic relations (7.6) the position and velocity components on the solid boundaries are calculated. Then the remaining boundary conditions are enforced

using either the explicit method of the implicit method of section 9.6.3. Now all nodes have well defined values for position, velocity, and density. Thus we can evaluate the various spacial derivatives needed for the governing equations (9.46)-(9.48). Finally the time derivative of the state variables $\dot{\mathbf{s}}_0$ is evaluated.

Having established $\mathbf{s}_0$ and $\dot{\mathbf{s}}_0$ we call the Runge–Kutta method like this

`rk23`(t_0, t_1, $\mathbf{s}_0$, $\dot{\mathbf{s}}_0$, $\mathbf{s}_1$, $\dot{\mathbf{s}}_1$)

where t_1 is the requested end time of the time step (this defines implicitly the time step as $\Delta t = t_1 - t_0$). Inputs are the state vector $\mathbf{s}_0$ and its time derivative $\dot{\mathbf{s}}_0$ at time t_0. This means that the routine `erk23` itself does not evaluate $\dot{\mathbf{s}}_0$. Outputs are the state vector $\mathbf{s}_1$ and its time derivative $\dot{\mathbf{s}}_1$ at time t_1. This means that `erk23` must evaluate the time derivative at time t_1, although that information is not needed to advance the solution for the current time step (see the previous section). There are, however, several good reasons for including the time derivatives at the end points of the time interval in the argument list for the time integrator

1. It is (still) easy to proceed to next time step by assigning

$$\begin{aligned} t_0 &:= t_1 \\ \mathbf{s}_0 &:= \dot{\mathbf{s}}_1 \\ t_1 &:= t_1 + \Delta t \end{aligned} \tag{9.137}$$

2. If a step fails, when using variable time stepping, $\dot{\mathbf{s}}_0$ does not need to be reevaluated by the time integrator.

3. We can use the same interface for either an explicit method (such as `erk23`) or an implicit method. This will be seen in the next section.

4. It is noted that the time integrator by itself only updates variables located in the state vector. However, we want to have the boundary values updated too, so that *all* variables are at $t = t_1$ upon return from `rk23`. This is accomplished simply by letting the method evaluate the time derivative at t_1 (this will require newly evaluated spacial derivatives, which rely on newly evaluated values on the boundary).

9.7.3 Implicit time integration

The previous section was devoted to an explicit time integration method. In this section we describe the implementation of an implicit Runge–Kutta method. The method under consideration is developed by Alexander [48] and features 3. order accurate advancement of the solution, 4. order embedded error estimation, stage

prediction through interpolation using Hermite polynomials, it has four stages, is singly diagonal implicit, and is L-stable.

In contrast to explicit Runge–Kutta methods, the implicit methods are characterized by having (one or more) *implicit* stage equations. In the most general setting the stage equations are fully coupled to each order. For the *diagonal implicit* methods, however, each stage equation only couples directly with itself. A great advantage of these methods over fully implicit methods is a computational saving. For general systems it is three times less costly to solve three systems of size $M/3$-by-$M/3$, than solving one system of size M-by-M. Specifically, we have for the k-th time step the following equations for the method under consideration

$$\begin{aligned}
\mathbf{s}_1 &= \mathbf{s}^k \\
\mathbf{s}_2 &= \mathbf{s}^k + h\left(A_{21}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + A_{22}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2\right) \\
\mathbf{s}_3 &= \mathbf{s}^k + h\left(A_{31}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + A_{32}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2) + A_{33}\mathbf{f}(t^k + c_3 h, \mathbf{s}_3)\right) \\
\mathbf{s}_4 &= \mathbf{s}^k + h\left(A_{41}\mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + A_{42}\mathbf{f}(t^k + c_2 h, \mathbf{s}_2) + A_{43}\mathbf{f}(t^k + c_3 h, \mathbf{s}_3)\right. \\
&\qquad \left. + A_{44}\mathbf{f}(t^k + c_4 h, \mathbf{s}_4)\right)
\end{aligned} \tag{9.138}$$

where matrix $\mathbf{A}$ holds the weights

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \alpha & \alpha & 0 & 0 \\ a_{31} & a_{32} & \alpha & 0 \\ b_1 & b_2 & b_3 & \alpha \end{bmatrix} \tag{9.139}$$

and vector $\mathbf{c}$ contains fractions of the time step

$$\mathbf{c} = \begin{pmatrix} 0 & 2\alpha & c_3 & 1 \end{pmatrix}^{\mathrm{T}} \tag{9.140}$$

and the constants are given by

$$\begin{aligned}
\alpha &= 1 - \frac{\sqrt{2}}{2}\cos\left(\frac{1}{3}\arctan\left(\frac{\sqrt{2}}{4}\right)\right) + \frac{\sqrt{6}}{2}\sin\left(\frac{1}{3}\arctan\left(\frac{\sqrt{2}}{4}\right)\right) \\
c_3 &= \frac{18}{13}\alpha^2 - 2\alpha + \frac{14}{13} \\
a_{31} &= -\frac{1}{4}\frac{-6\alpha c_3 + c_3^2 + 4\alpha^2}{\alpha} \\
a_{32} &= -\frac{1}{4}\frac{c_3(2\alpha - c_3)}{\alpha} \\
b_1 &= -\frac{1}{12}\frac{-18\alpha c_3 + 12\alpha^2 c_3 + 3c_3 + 12\alpha - 12\alpha^2 - 2}{\alpha c_3} \\
b_2 &= \frac{1}{12}\frac{-3c_3 + 6\alpha c_3 - 6\alpha}{\alpha(2\alpha - c_3)} \\
b_3 &= -\frac{1}{3}\frac{6\alpha^2 + 1 - 6\alpha}{c_3(2\alpha - c_3)}
\end{aligned} \tag{9.141}$$

From this we see that the first stage equation is explicit – the value is given directly by the solution from the previous time step. The second stage equation couples only with itself, since $\mathbf{s}_1$ is explicitly known. Thus we may solve for $\mathbf{s}_2$ independent of $\mathbf{s}_3$ and $\mathbf{s}_4$. Once $\mathbf{s}_2$ is found we proceed with the third stage equation, which again only couples directly with itself. Finally $\mathbf{s}_4$ is calculated implicitly, using the known values of $\mathbf{s}_1$, $\mathbf{s}_2$, and $\mathbf{s}_3$. This solution procedure is possible because the method is only diagonally implicit.

The new solution $\mathbf{s}^{k+1}$ is given simply by the final stage value $\mathbf{s}_4$, that is

$$\mathbf{s}^{k+1} = \mathbf{s}_4 \tag{9.142}$$

An advantage of this is that the time derivative $\dot{\mathbf{s}}_4$, which is calculated during the solution phase, can be reused as the time derivative of $\mathbf{s}_1$ for the following time step. That is

$$\mathbf{f}(t^{k+1}, \mathbf{s}_1) = \dot{\mathbf{s}}_4^k \tag{9.143}$$

where k denotes positions in time $t = t^k$, $k = 1, 2, \ldots$ The solution is advanced with 3. order accuracy.

The error estimator is based on an embedded 4. order solution, which is given by

$$\begin{aligned}\tilde{\mathbf{s}}^{k+1} = \mathbf{s}^k + h \Big(&\tilde{b}_1 \mathbf{f}(t^k + c_1 h, \mathbf{s}_1) + \tilde{b}_2 \mathbf{f}(t^k + c_2 h, \mathbf{s}_2) + \tilde{b}_3 \mathbf{f}(t^k + c_3 h, \mathbf{s}_3) \\ &\tilde{b}_4 \mathbf{f}(t^k + c_4 h, \mathbf{s}_4) \Big)\end{aligned} \tag{9.144}$$

where

$$\begin{aligned}\tilde{b}_1 &= \frac{1}{24} \frac{12\alpha c_3 - 4\alpha - 2c_3 + 1}{\alpha c_3} \\ \tilde{b}_2 &= \frac{1}{24} \frac{-1 + 2c_3}{\alpha(2\alpha - c_3)(2\alpha - 1)} \\ \tilde{b}_3 &= -\frac{1}{12} \frac{-1 + 4\alpha}{c_3(2\alpha - c_3)(c_3 - 1)} \\ \tilde{b}_4 &= \frac{1}{12} \frac{3 + 12\alpha c_3 - 4c_3 - 8\alpha}{(2\alpha - 1)(c_3 - 1)}\end{aligned} \tag{9.145}$$

The strategy for variable time stepping follows the same lines as in the section 9.7.2. The only difference is that we have fourth order error estimation so (9.135) is replaced by the following formula for the time step

$$h_{\text{new}} = r h_{\text{cur}} \left(\frac{\varepsilon}{||\mathbf{e}||} \right)^{1/4} \tag{9.146}$$

where

h_{new} step size for the next time step
h_{cur} the current step size
$||\mathbf{e}||$ 2-norm of the error vector $(\mathbf{s}^{k+1} - \tilde{\mathbf{s}}^{k+1})$
ε allowed error (9.136)
$r = 0.95$ safety factor

Solving the stage equations

The generic form of one stage equation is given by

$$\mathbf{s}_i = \mathbf{s}^k + h \sum_{j=1}^{4} A_{ij} \mathbf{f}(t^k + hc_j, \mathbf{s}_j) \tag{9.147}$$

defined for $i = 2, 3, 4$ (the first stage is explicit). In general each stage equation is nonlinear (because $\mathbf{f}$ is nonlinear) and must be solved by iteration. The typical choice is to use (modified) Newton-Raphson iterations. In order to do this we write (9.147) in residual form and establish the Jacobian

$$\begin{aligned} \mathbf{r}_i &= \mathbf{s}^k - \mathbf{s}_i + h \sum_{j=1}^{4} A_{ij} \mathbf{f}(t^k + hc_j, \mathbf{s}_j) \\ \mathbf{M} &= \frac{\partial \mathbf{r}_i}{\partial \mathbf{s}_i} = -\mathbf{I} + hA_{ii} \mathbf{J}(t^k + hc_i, \mathbf{s}_i) \end{aligned} \tag{9.148}$$

where we have used the notation

$$\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{s}} = \begin{bmatrix} \dfrac{\partial f_1}{\partial s_1} & \dfrac{\partial f_1}{\partial s_2} & \cdots \\ \dfrac{\partial f_2}{\partial s_1} & \dfrac{\partial f_2}{\partial s_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \tag{9.149}$$

Because $A_{ii} = \alpha$ for $i = 2, 3, 4$ it is clear that the iteration matrix $\mathbf{M}$ does not change between stages if $\mathbf{J}$ is constant. This is very attractive since we can then employ modified Newton-Raphson iterations, keeping the iteration matrix constant for a number of iterations also between stages.

The Jacobian matrix $\mathbf{J}$ is formed by taking the gradient of each of the equations of the system (9.124). Each column correspond to the gradient with respect to a specific state variable s_i. It is noted that $\mathbf{s}$ also contains the positions of the computational nodes due to the Lagrangian formulation. This is unusual compared to the normal situation where the position of the nodes are fixed, and therefore do not enter the state vector.

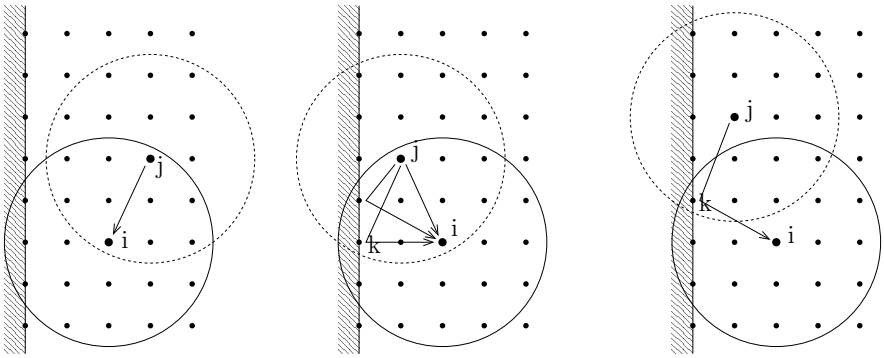

Figure 9.14: Coupling with boundary nodes.

The gradients of the governing equations are given in section 9.5.1. Special care is necessary when evaluating terms such as $\mathrm{d}(\partial u_i/\partial x)/\mathrm{d}\rho_j$. For a node i with no interaction with nodes on the boundary this term will be zero. If interaction with boundary nodes is present the situation is more complicated. Three different situations are shown in figure 9.14.

1) The first subfigure (left) shows a situation where a perturbation of node j will cause a change in the functions (i.e. the spacial derivatives of u, v, and ρ) at node i.

2) In the middle subfigure we have again a direct influence on the functions evaluated at node i from a perturbation of node j. However, in this case a secondary effect also takes place. The values on the boundary depend on node j (because node j is within the smoothing length) and therefore the values on the boundary at e.g. node k are sensitive to a perturbation at node j. But node i has boundary node k within its smoothing length, so changing a value at node k will affect the functions evaluated at node i. These dependencies are shown graphically by arrows, and must be taken into account using the chain rule when computing the sensitivity at node i with respect to node j.

3) Finally, the right subfigure of figure 9.14 shows a situation, where a node j outside the smoothing length of node i affects the functions evaluated at node i, through a coupling with nodes on the boundary. These effects must be taken into account when evaluating the sensitivities at nodes close to the boundary.

Including all terms we get the following general expression for the sensitivities at a node i with respect to changes in node j

$$\frac{\mathrm{d}}{\mathrm{d}(\cdot)_j}\left(F_i\right) = \frac{\partial F_i}{\partial(\cdot)_j} + \sum_{k=1}^{N_\mathrm{b}} \left(\frac{\partial F_i}{\partial u_k}\frac{\partial u_k}{\partial(\cdot)_j} + \frac{\partial F_i}{\partial v_k}\frac{\partial v_k}{\partial(\cdot)_j} + \frac{\partial F_i}{\partial \rho_k}\frac{\partial \rho_k}{\partial(\cdot)_j} \right) \tag{9.150}$$

where F_i is either of the spacial derivatives of u, v, ρ with respect to x or y, and $(\cdot)$ is either of x, y, u, v, or ρ, and N_b is the number of boundary nodes

within the smoothing length of node i. The first term on the right hand side of (9.150) is given directly by the sensitivities in (9.40) and (9.44), and corresponds to the direct coupling between node i and node j. The terms $\partial F_i/\partial u_j$, $\partial F_i/\partial v_j$, and $\partial F_i/\partial \rho_j$ are also given by (9.40) and (9.44). Finally, the terms $\partial u_k/\partial(\cdot)_j$, $\partial v_k/\partial(\cdot)_j$, and $\partial \rho_k/\partial(\cdot)_j$ are given by the sensitivity analysis of the boundary conditions in section 9.6.4. Combining all of these terms allows us to evaluate the Jacobian matrix of the system analytically. It should be noted that the terms in the summation of (9.150) are not non-zero at all times. Depending on the context some of the terms are identically zero. This specialization of (9.150) has been carried out during the implementation in the simulation program. We do not display the details in this text.

It should be noted that some of the gradients related to triple points are currently not implemented. In fact the derivatives of U^i_{x}, U^i_{y}, and U^i_{f} with respect to x_j and y_j are *not* implemented in the simulation program. Therefore, we cannot obtain the Jacobian matrix using the analytical approach if the simulation problems contains triple points. These missing expressions could added as a part of the future work.

As mentioned earlier we solve the stage equations by simplified Newton-Raphson iterations. A means to speed up the convergence of the stage value iterations, is to employ *stage prediction*. The idea is to provide an improved starting value for the simplified Newton-Raphson procedure. Especially when the iteration matrix is not updated in every iteration, a good starting point can cut down the number of iterations by a significant amount. Prediction of the stage value can be accomplished in different ways. We have followed the method given by Alexander, which is based on local information within the current time step. The first stage value $\mathbf{s}_1$ is given explicitly, as we have already seen. The second stage value is located at $c_2 \approx 0.872$ (fraction of the time step) and is predicted by a first order extrapolation from $\mathbf{s}_1$ using the gradient information at $c_1 = 0$. The third stage value is located at $c_3 \approx 0.468$, that is in between the two first stage values. The value of $\mathbf{s}_3$ is interpolated using Hermite polynomials fitted between c_1 and c_2 (a Hermite polynomial is a cubic polynomial, which matches function values and slopes at the end points). Finally, the last stage value is located at $c_4 = 1$, and the value is predicted by extrapolation using the same Hermite polynomials as before. This means that the true value of $\mathbf{s}_3$ is not used for the prediction of $\mathbf{s}_4$.

9.8 Remeshing

In this section we discuss how to redistribute the computational nodes during a simulation. There are two reasons why this is necessary 1) distortion of the nodal layout, 2) uncontrolled modes must be handled.

1) Due to the Lagrangian formulation the computational nodes must be moved

along the velocity field of the fluid flow. In some sense this is attractive from the point of view of automatic adaptability of the nodal layout. However, in general we cannot rely on this "adaptation" alone. In fact it is necessary to *redistribute* the nodes, or using a familiar notion *remesh* the system (we use this word, although we do not predefine the connectivity between nodes). This is so because there will in general be locations of stagnation flow where nodes would cluster. In other areas the resolution would become more and more coarse due to a spreading flow. In a way we might say that the adaptive nature of the Lagrangian approach is at one time beneficial, and at the other time problematic.

2) The spacial discretization method is based on the Moving Least Squares method, which can be interpreted as a generalization of the finite difference method. Using a term from finite differences, we may characterize our discretization method as being *collocated*. Collocated means that the field variables u, v, and ρ are defined at identical points in space, as opposed to a *staggered* layout in which some nodes carry e.g. velocity components and other nodes carry density (or pressure). It is well known that a collocated layout can suffer from *zero energy modes*. A zero energy mode is a mechanism, which leaves the spacial derivatives unchanged when a field variable is perturbed.

Three examples of zero energy modes are sketched in figure 9.15 for a 1D situation. The top left subfigure shows the so-called "checker-board" decoupling. The oscillatory behavior of the field variable is being translated into a zero-field for the first derivative. This observation has lead to the notion of aliasing of a high frequency component. In the top right subfigure another example is shown. Again the nodal layout is even, and because of this the center node, does not enter the expression for the first derivative. Clearly, a perturbation of the center node will not change the slope at the center node – a decoupling effect. Finally, in the lower subfigure of figure 9.15 we have shown an example with uneven nodal layout.

Although the situation is greatly improved by uneven distribution of the nodes, there is still a possibility of high frequency being interpreted as zero frequency. The following sources for triggering of inadmissible modes exist 1) too large time steps, 2) numerical truncation errors (finite precision of the computer), 3) introduction of kinematic constraints on the field variables, and 4) discontinuities in the field variables. Too large time step should in general be avoided, since it affects stability and accuracy of the solution. Numerical truncation errors cannot be avoided, so for this reason the discretization method should be robust with respect to perturbations. Kinematic constraints and discontinuities are in some sense alike. In both cases special care is needed.

When the approximation of derivatives is based on finite differences there are basically three different methods available for overcoming the problem of zero energy modes 1) asymmetric formula for the first derivative, 2) staggered variables, 3) filtering of the field variable. We have chosen the last possibility, since we can get it "for free" through the remeshing procedure. When the nodes

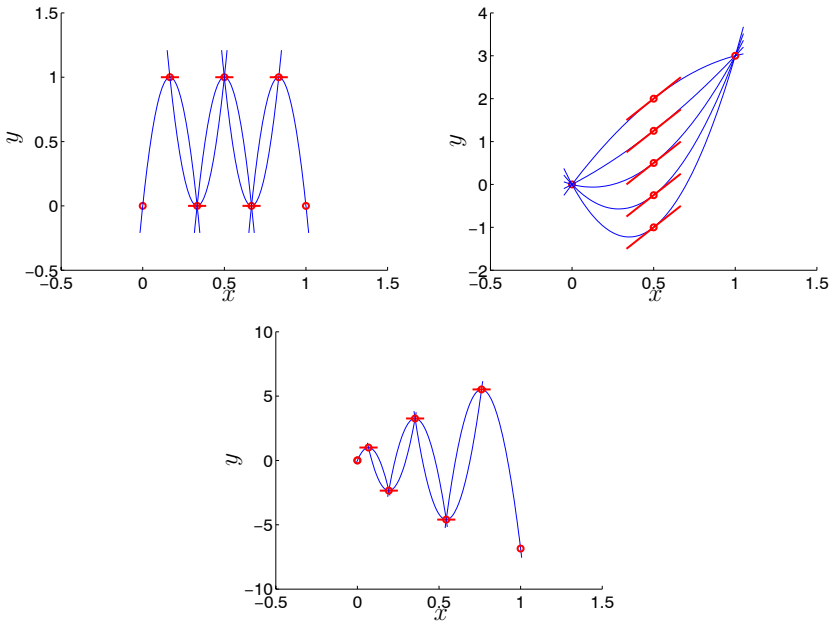

Figure 9.15: Top left: The well-known "checker board" decoupling. Top right: Decoupling of center node. Bottom: Checker board with uneven nodal layout.

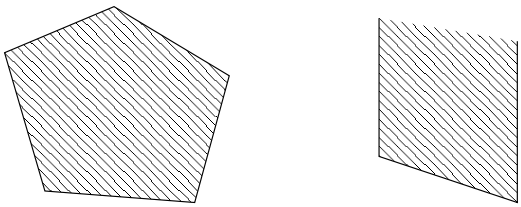

Figure 9.16: Left: A closed body. Right: An open body.

are redistributed the field variables must be *interpolated* onto the newly defined nodal positions. This interpolation has a filtering effect, which can be utilized to avoid growing decoupling in the system.

In the following we show how the remeshing of the model is done. We treat solids, free surfaces, and the fluid domain in individual subsections. In each case we describe how to find the *positions* for the remeshed nodes. After this a section on how to interpolate the field values u, v, and ρ at the new nodal positions is given. The treatment of zero energy modes is not discussed further.

9.8.1 Remeshing of solids

In this work we consider only rigid bodies with boundaries consisting of line segments. One can think of a polygon with n edges, although the bodies we consider do not have to be closed. In figure 9.16 an example of both types of bodies is shown.

The presence of a body is only relevant as a boundary condition for the fluid domain. Therefore we only put computational nodes on the part of a body that is touching the fluid domain. Depending on the situation the body might be fully immersed in the fluid, or only partly immersed. A graphical picture of this is shown in figure 9.17. For a partly immersed body we need an "end point" in order to define the transition between the wetted and non-wetted part of the body. This point can be either 1) a triple point, or 2) given implicitly (this will be explained shortly).

1) The first possibility is used to describe the contact point between a free surface and a body. Since we do not model "air" it is not necessary to have computational nodes on a body outside a triple point. See figure 9.17 (right subfigure).

2) The second possibility is used if less than two triple points are specified. In this case the program will assume the wetted region to begin (end) at the first (last) point used to define the geometry of the body. The program can handle different topologies according to the following summary

1. body is open and one or two triple points are given

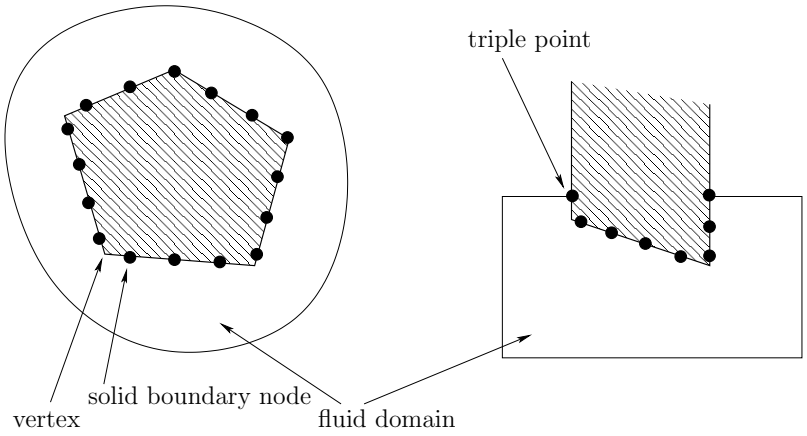

Figure 9.17: Left: A fully immersed body. Right: A partly immersed body.

2. body is closed and no triple points are given

The limits for the wetted part on the body is represented using the curve length along the boundary of the body, and are denoted by τ_{begin} and τ_{end}. For example the location of a triple point is in general not constant, so the corresponding coordinate measured along the body (τ_{begin} or τ_{end}) will also change. Based on the upper and lower limit for the wetted region the computational nodes are distributed in an evenly manner. The number of nodes is found from

$$N = \texttt{floor}(\frac{\tau_{\text{end}} - \tau_{\text{begin}}}{\Delta x}) \tag{9.151}$$

where Δx is the desired nodal spacing. Because N must be an integer the desired nodal spacing will be achieved only approximately. The nodes are located on the line segments at the locations corresponding to the curve length coordinates given by

$$\tau_{\text{new}} = \tau_{\text{begin}} + \widetilde{\Delta x}(i-1), \quad i = 1, 2, \cdots, N \tag{9.152}$$

where $\widetilde{\Delta x} = (\tau_{\text{end}} - \tau_{\text{begin}})/(N-1)$. Figure 9.18 shows an example of how the computational nodes are distributed.

In general it is necessary to repeat the remeshing of a solid if at least one of the limits of the wetted region is governed by a triple point. This is because the wetted region in general will change its location and/or size. So basically we have to repeat the remeshing at every time step. In fact, to be precise, we repeat it also at the substeps of a time integration method.

It should be mentioned that a triple point should never separate from the solid boundary with which it is associated. However, the position of a solid boundary

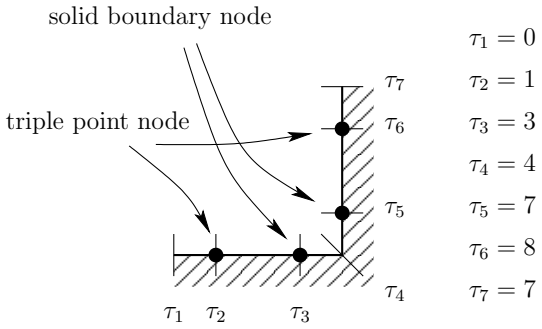

Figure 9.18: An example of the nodal distribution on a solid. The discretization length is $\Delta x = 2$.

is given by prescribed kinematics, while the position of a triple point is evolved by time integration of its velocity components. Due to integration errors the triple point will little by little no longer coincide with a line segment on the solid boundary. In order to avoid this kinematic violation we invoke a projection of the triple point onto the solid boundary. This is done in every time step (or substep). The projection is very simple and works by finding the point on a line segment with the shortest distance from the triple point. This is sketched in figure 9.19.

9.8.2 Remeshing of free surfaces

In this work we describe free surfaces by a collection of points. Two kinds of free surfaces are allowed 1) closed surfaces, and 2) open surfaces. In figure 9.20 an example of both types is shown. If a free surface is open it must be ended by triple points.

The initial shape of a free surface is given by the definition of the problem under consideration. The nodes on the free surface are ordered so that connecting the nodes with line segments from node 1 to 2, and from node 2 to 3, etc, would become a meaningful boundary for the fluid domain. Thanks to the Lagrangian formulation the evolution of the free surface is obtained directly. However, as previously mentioned, the flow field might cluster the nodes in some areas on the free surface (or spread the nodes) making the discretization uneven. Also the movement of solid bodies might change the extent of the free surface dramatically. An example of this is shown in figure 9.21. It is noted that the free surface is distinct from the interior of the fluid domain. This means that nodes belonging to the interior of the fluid domain can never become a part of the free surface. Similarly a node on the free surface remains a free surface node.

Let us assume that a free surface needs to be remeshed. First we calculate the curve length along the line segments connecting node 1 to node 2, node 2 to node 3, and so on. Then the number of nodes N for the new discretization is

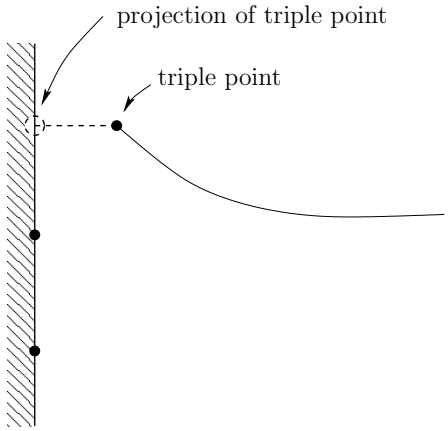

Figure 9.19: Projection of a triple point onto a solid boundary (separation from solid boundary is exaggerated).

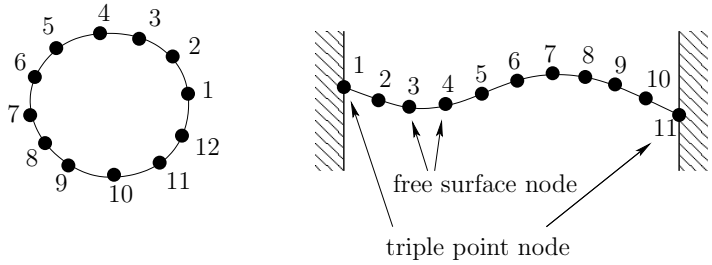

Figure 9.20: Left: A closed free surface. Right: An open free surface. Each node carries an index in an ordered manner as indicated.

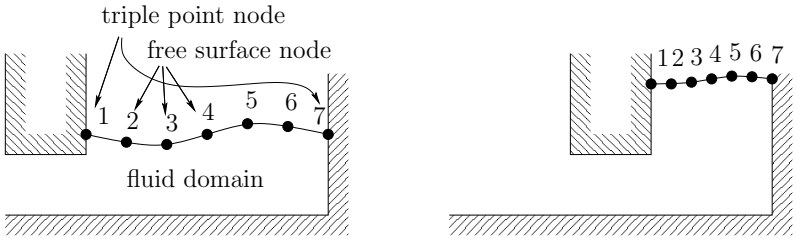

Figure 9.21: Change in extent of free surface due to movement of solid boundary.

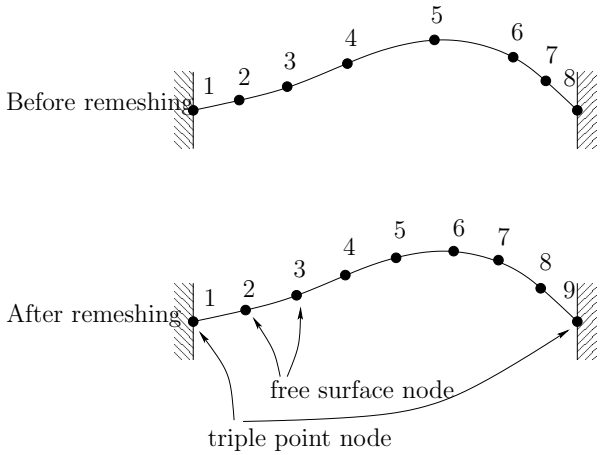

Figure 9.22: An example of remeshing of a free surface.

calculated by

$$N = \texttt{floor}(\frac{\tau_{\text{end}} - \tau_{\text{begin}}}{\Delta x}) \tag{9.153}$$

where $\tau_{\text{begin}} = 0$, τ_{end} is the total length along the line segments, and Δx is the desired node spacing. The new nodes should then be located at positions

$$\tau_{\text{new}} = \widetilde{\Delta x}(i-1), \quad i = 1, 2, \cdots, N \tag{9.154}$$

where $\widetilde{\Delta x} = \tau_{\text{end}}/(N-1)$. In order to find the coordinates $(x_{\text{new}}, y_{\text{new}})$ corresponding to τ_{new} we use interpolation based on the MLS method. We use a 1D polynomial basis given by

$$\begin{aligned} p_1(x) &= 1 \\ p_2(x) &= x \\ p_3(x) &= x^2 \end{aligned} \tag{9.155}$$

Using data points $(\tau_1, x_1), (\tau_2, x_2), \ldots$ from the current free surface and the MLS method, we can calculate new positions $x_1^{\text{new}}, x_2^{\text{new}}, \ldots$ A similar procedure is used for the y-positions.

I figure 9.22 we show an example of the nodal layout before and after remeshing. It is noted that the first and the last node on the free surface are special. If the free surface is open (ended by triple points) then we do not modify the position of the first and the last node. If the free surface is closed the last node *is* remeshed, so only the first node remains where it were.

The free surfaces are remeshed from time to time, typically at fixed intervals. In contrast with the solids we do not allow remeshing of the free surfaces at the

substeps of the time integration method. The reason for this is that it would be difficult to handle a possible change in the number of nodes on the free surface (this would also mean a change in the size of the state vector).

9.8.3 Remeshing of the fluid domain

The initial layout of the nodes in the interior of the fluid domain is given by the specification in the problem definition. After some time steps the nodes need to be redistributed in order to prevent clustering of nodes or void areas with no nodes. In some cases it is possible to perform the remeshing in a very simple manner. If for example one is solving the lid driven cavity problem, then the nodal positions can be reinitialized to the initial (typically regular) layout. If on the other hand the domain of the fluid changes as the simulation proceeds, then a general method must be used.

Several ways of remeshing a two dimensional domain exist. For example, one method is based on solving a Poisson equation, while another method calculates a triangulation of the domain. In this project we have tried to use what seems to be a very simple method, namely, an artificial potential method.

At the time of remeshing the idea is to let all nodes carry a potential. The interior nodes are then allowed to move, while the nodes on the boundary remain fixed. Ideally this system has a configuration of minimal "energy" in which all nodes share the same average distance to each other. In practice, however, it was found that the steady state configuration was hard to find. In fact it would be necessary to carry out something like an SPH simulation. This would of course work, but only at a significant cost, because the simulation would need several time steps before the excess of potential energy was dissipated through viscosity.

As an attempt to develop a fast remeshing algorithm for arbitrary domains the following strategy was followed.

First we choose as a potential a spring with undeformed length greater than zero. This gives a force along the spring given by

$$F = k(L - L_0) \tag{9.156}$$

where

F	force
k	coefficient of stiffness
L	deformed length
L_0	undeformed length

In figure 9.23 we have sketched two nodes connected by a spring. Referring to the figure we now develop an expression for the force on node i due to the spring. From ratios of triangles we get

$$\frac{F_x^{ij}}{x_j - x_i} = \frac{F^{ij}}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}} \tag{9.157}$$

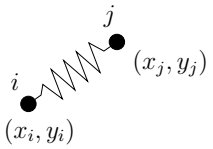

Figure 9.23: Two nodes connected by a spring.

or using (9.156) and rearranging (and giving the corresponding expression for the y-component)

$$\begin{aligned} F_x^{ij} &= k\left(1 - \frac{L_0}{r_{ij}}\right)(x_j - x_i) \\ F_y^{ij} &= k\left(1 - \frac{L_0}{r_{ij}}\right)(y_j - y_i) \end{aligned} \tag{9.158}$$

where $r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. If we loop over the neighbors of node i and sum the contributions we get the total force on node i

$$\begin{aligned} F_x^{i} &= \sum_j k\left(1 - \frac{L_0}{r_{ij}}\right)(x_j - x_i) \\ F_y^{i} &= \sum_j k\left(1 - \frac{L_0}{r_{ij}}\right)(y_j - y_i) \end{aligned} \tag{9.159}$$

However, we do not want to include neighbors that are farther away than some cut off distance. We define Δx as the desired average nodal spacing, and from this we define the cut off distance as

$$r_{\text{cut}} = 1.1\Delta x \tag{9.160}$$

Thus, if a neighbor has $r_{ij} > r_{\text{cut}}$ then we exclude it from the sum of forces. Since we want Δx to be the average spacing between nodes we use this value for L_0, that is $L_0 = \Delta x$. In this way nodes that are too close will repeal each other, while nodes that are to far away (but within the cut off distance) will attract each other.

Finally, we update the positions one node at a time. This means that we try to find a zero of (9.159) for node i, while all other nodes are fixed. It is noted that (9.159) is a nonlinear function because $L_0 \neq 0$. Therefore iteration is needed,

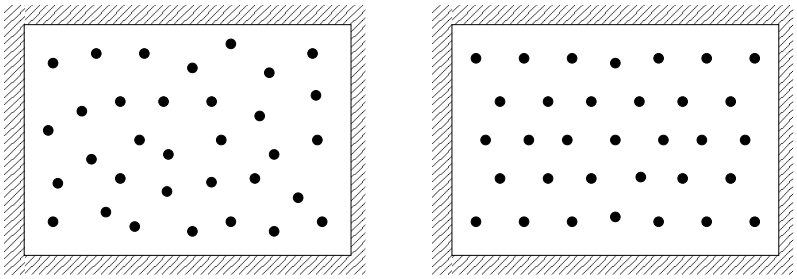

Figure 9.24: An example of nodal distribution before (left) and after (right) remeshing.

which is done by taking Newton steps. The gradient of (9.159) is given by

$$\begin{aligned}
\frac{\partial F_x^i}{\partial x_i} &= \sum_j \left(\frac{\partial F^{ij}}{\partial x_i}(x_j - x_i) - F^{ij} \right) \\
\frac{\partial F_x^i}{\partial y_i} &= \sum_j \frac{\partial F^{ij}}{\partial y_i}(x_j - x_i) \\
\frac{\partial F_y^i}{\partial x_i} &= \sum_j \frac{\partial F^{ij}}{\partial x_i}(y_j - y_i) \\
\frac{\partial F_y^i}{\partial y_i} &= \sum_j \left(\frac{\partial F^{ij}}{\partial y_i}(y_j - y_i) - F^{ij} \right)
\end{aligned} \tag{9.161}$$

where

$$\begin{aligned}
\frac{\partial F^{ij}}{\partial x_i} &= -\frac{\partial F^{ij}}{\partial r_{ij}} \frac{x_j - x_i}{r_{ij}} = -\frac{L_0(x_j - x_i)}{(r_{ij})^3} \\
\frac{\partial F^{ij}}{\partial y_i} &= -\frac{\partial F^{ij}}{\partial r_{ij}} \frac{y_j - y_i}{r_{ij}} = -\frac{L_0(y_j - y_i)}{(r_{ij})^3}
\end{aligned} \tag{9.162}$$

The reason for updating the position for one node at a time is that it makes the algorithm more stable. Actually, it will converge only slowly if all nodes were updated at once. In that case, we found it necessary to reduce the size of the Newton step in order to avoid divergence. In figure 9.24 we show an example of the positions of the computational nodes before and after remeshing.

The performance of the suggested remeshing method is fair but not fail proof. If the fluid has a main direction of flow, e.g. so that the nodes all travel from the left to the right, then the remeshing algorithm might not be able to reposition the nodes as far upstream as it actually should. Thus there will be a net movement of the nodes with the flow. This can be a disadvantage.

As for the free surface, the fluid domain is remeshed at regular intervals, but never during a substep of the time integration routine.

9.8.4 Interpolation of field variables

Once the new locations of the computational nodes have been found we need to interpolate the field values from the current nodes to the new nodes. How this is done depends on what part of the computational domain is considered.

Solid boundaries are remeshed at every (sub)step of the simulation. Positions and velocities are calculated directly from the kinematics of the solid body. The density is evaluated as a part of the boundary conditions. This means that at the end of a time step *all* values on solid boundaries have been remeshed and are up to date (remember that the time integrator evaluates the time derivative of the state vector also at the end of the time step – this implies an evaluation of the boundary conditions).

Nodes on free surfaces and the interior of the fluid domain are treated alike. Once the new positions of the nodes have been found we loop over free surface nodes and interior nodes one by one, and perform an interpolation of u, v, ρ from the current field variables. So two copies of the system exists at this point, one holding the current nodes and the other holding the new nodes. When the interpolation step is completed the current nodes are modified according to the new values (note that the total number of nodes may grow or shrink or remain unchanged on each free surface and the interior). Finally, the spacial derivatives are reevaluated based on the new field variables and the time derivative of the state vector is calculated. Having reestablished a state vector and its time derivative we are ready to proceed to the next time step.

The important feature is that the boundary conditions are *not* enforced before evaluation of the spacial derivatives. If we use the Neumann condition for density, this would require interpolation of the "old" velocity components used to approximate the acceleration of a node (see section 8.8). Interpolation of the "old" velocities is critical, or rather problematic, as can be seen below.

The approximation of the x-acceleration of a node on a free surface is given by (9.66)

$$a_x^i = \frac{u_i - u_i^0}{t - t^0} \tag{9.163}$$

Here u_i^0 is the u-velocity at the beginning of the current time step $\Delta t = t - t^0$ (the very first time step is special as discussed in section 8.8). Clearly, if we want to impose the boundary conditions after remeshing of the system, then we would also need remeshed values of u_i^0. This can of course be done, but the problem is that if some interpolation error E exists, then (9.163) becomes more and more inaccurate as the time step is decreased. For example, a Δt of the magnitude 10^{-5} and a true acceleration a_x^i of size 10 would require an interpolation error no bigger than say 10^{-6}. Such accuracy is not easy to obtain, and the problem gets even worse when Δt and the magnitude of u_i decreases.

So, if we should decide to re-impose the boundary conditions after remeshing an error would occur at free surfaces (when using the Neumann condition for

density). This error would then propagate to the spacial derivatives close to the boundary.

However, we do not need to re-impose the boundary conditions after remeshing (because u, v, and ρ are already known at *every* node we can evaluate the spacial derivatives directly). Choosing not to do so eliminates the need of interpolating the "old" velocity components on free surfaces, and eliminates the induced error it would imply.

Chapter 10

Results from the Navier–Stokes equations

In this chapter we report results from the simulation program for the Navier-Stokes equations.

We begin by solving a few test problems in order to validate the code and establish the convergence properties. We also monitor the allowed step size for time integration and the computation time spent (using the computers wall clock). The test problems are 1) lid driven cavity, 2) fixed incline slider bearing, 3) rotating free surface, and 4) drop oscillations.

After the test problems we carry out two simulations of piston ring lubrication. The first problem tries to solve the problem in a fully transient manner, taking full advantage of the Lagrangian formulation in order to handle the solid-fluid interaction. The second problem models only the part of the oil film close to the inlet of the piston ring. In this case the Reynolds equation is used to establish boundary conditions at the outlet of the simulation domain.

10.1 Lid driven cavity

The lid driven cavity problem is a standard benchmark for flow solvers. The simulation domain consists of a square region confined by solid walls. The walls are stationary, with exception of the top wall which is moving at a constant horizontal speed. As a consequence a singularity develops at the top left and right corner, where the x-component of velocity is not single valued.

We have solved the lid driven cavity problem using the following parameters

$$\begin{aligned} L &= 1 \quad \text{side length of square, m} \\ u &= 1 \quad \text{horizontal velocity of lid, m/s} \\ \mu &= 1 \quad \text{dynamic viscosity, Pa s} \\ \rho_0 &= 1 \quad \text{initial density, kg/m}^3 \\ p_0 &= 101500 \quad \text{reference pressure at } \rho = \rho_0, \text{ Pa} \\ \text{time integrator} &\quad \text{Explicit 3rd order Runge Kutta} \\ \text{time step} &\quad \text{Fixed} \\ \text{eos} &\quad \text{Dowson-Higginson} \\ \text{density bc} &\quad \text{Neumann condition} \end{aligned} \tag{10.1}$$

For the equation of state we use the Dowson-Higginson relation (6.15). Note that the reference pressure p_0 enters as a parameter for the equation of state although the driven cavity is fully confined by walls. This gives a speed of sound of approximately $a = 355$ m/s, and thus a Mach-number of around $Ma = 0.0028$. At the same time the Reynolds number is $Re = 1$, so basically the flow can be regarded as almost incompressible and in the creeping flow limit. Thus we should be able to compare the results from the Navier–Stokes equations with a reference solution based on a creeping flow formulation.

We engage the remeshing procedure at every 10 time step for all simulations (i.e. no matter the size of the time step). In order to facilitate the comparison of results the nodes are remeshed to a regular grid layout, and further more we always invoke remeshing after the final time step. The evolution of the flow field is shown in figure 10.1 for $N = 31$.

In figure 10.2 we have plotted the x-component of velocity along the line $x = L/2$, together with a solution obtained from an incompressible creeping flow model.[1] In both cases $N = 201$. It is seen that the results are very similar as expected.

In table 10.1 we have shown the simulation results for various discretizations. The first column holds the number of discretization points N in horizontal and vertical direction including nodes on the walls. The second column holds the spacial discretization Δx given by

$$\Delta x = \frac{L}{N-1}$$

The third column contains the maximum allowed time step Δt (fixed throughout the simulation). The two remaining columns hold quantities used to monitor the convergence properties 1) the integral I_u of u along $x = L/2$, 2) the center

[1] The creeping flow model was implemented during the one semester course 41319 Computational Fluid Dynamics at the Technical University of Denmark.

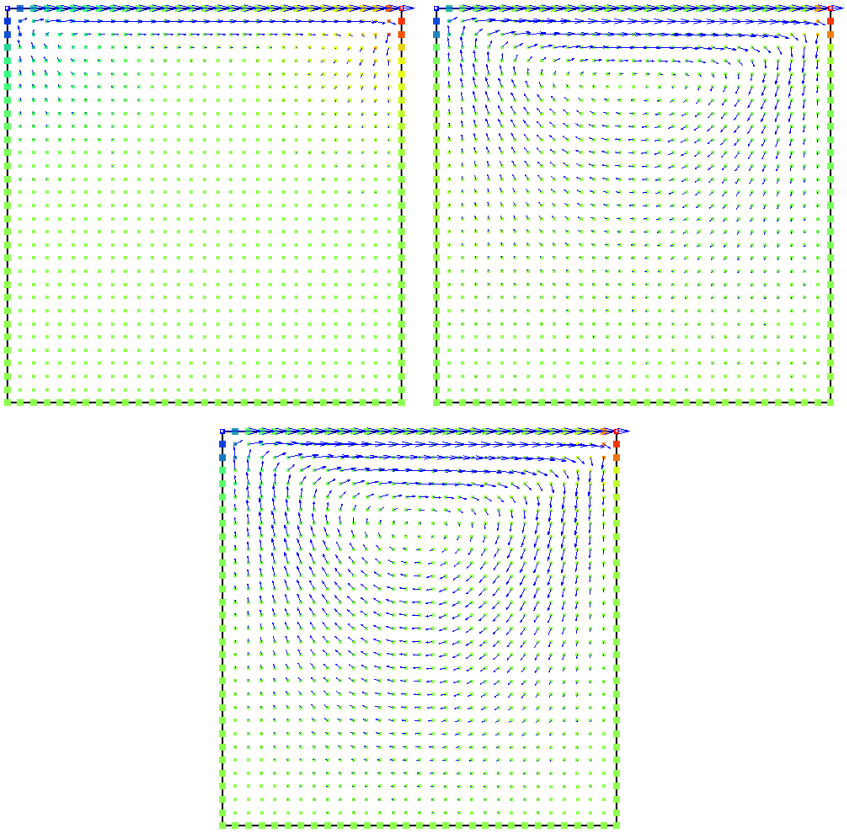

Figure 10.1: Evolution of the flow field of the lid driven cavity. Red color means high density, while blue color denotes low density. The velocity vectors are indicated by arrows. Top Left: $t = 1.2 \cdot 10^{-3}$ s, Top Right: $t = 1.2 \cdot 10^{-2}$ s, Bottom: $t = 0.1$ s (converged).

N	Δx [m]	Δt [s]	I_u [m^2/s]	u_{center} [m/s]	t_{elapsed} [s]
11	1.000E-1	6.8E-4	5.29E-2	-1.49E-1	1.937
31	3.333E-2	2.4E-4	1.87E-2	-1.83E-1	23.969
51	2.000E-2	1.5E-4	1.11E-2	-1.92E-1	94.266
71	1.143E-2	1.1E-4	7.68E-3	-1.96E-1	244.047
91	1.111E-2	9.1E-5	6.07E-3	-1.98E-1	472.500
201	5.000E-3	2.1E-5	2.66E-3	-2.02E-1	9719.060

Table 10.1: Simulation results from the lid driven cavity problem.

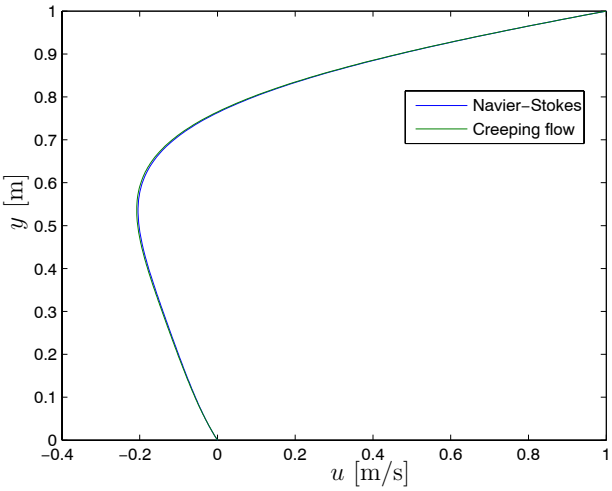

Figure 10.2: Solutions for the lid driven cavity problem obtained from 1) the Navier–Stokes equations (compressible, $Ma \approx 0.0028$, $Re = 1$), and 2) an incompressible creeping flow model. The number of discretization points along each axis is $N = 201$.

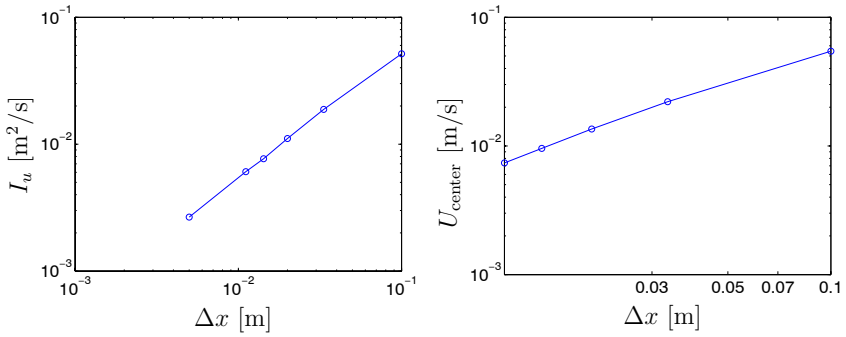

Figure 10.3: Convergence rate of the lid driven cavity problem. Left: I_u plotted against Δx, Right: u_{center} plotted against Δx.

node u-velocity at steady state. At steady state I_u should be zero in order to satisfy continuity. The center u-velocity at steady state is unknown, but can be estimated from a high resolution solution of the problem. Thus we calculate the error in u_{center} from

$$E = |u^N_{\text{center}} - u^{301}_{\text{center}}| \tag{10.2}$$

which we evaluate for $N = 11, 31, 51, 71, 91$. Note that $N = 201$ must be discarded from this investigation, since $N = 301$ is not a high resolution compared with $N = 201$. Note also that we have taken $u^{301}_{\text{center}} = -0.205$ m/s from the creeping flow solution of the problem. In figure 10.3 we have plotted I_u and E against the spacial resolution Δx. Using the Matlab function `polyfit` we are able to compute the slope of the (almost) linear curves by fitting a straight line in the least squares sense. The result is

$$\begin{aligned} \alpha_I &= 1.0001 \\ \alpha_u &= 0.9204 \end{aligned} \tag{10.3}$$

These figures indicate that the spacial discretization has first order convergence. This result could be anticipated from the knowledge of convergence rates for 1D finite differences. For arbitrary nodal spacings the following differences develop for the first and second derivative, respectively [8]

$$\begin{aligned} \left(\frac{\mathrm{d}u}{\mathrm{d}x}\right)_{\mathrm{P}} &= \frac{1}{\Delta x_{\mathrm{W}} + \Delta x_{\mathrm{E}}} \left(\frac{\Delta x_{\mathrm{W}}}{\Delta x_{\mathrm{E}}}(u_{\mathrm{E}} - u_{\mathrm{P}}) + \frac{\Delta x_{\mathrm{E}}}{\Delta x_{\mathrm{W}}}(u_{\mathrm{P}} - u_{\mathrm{W}}) \right) \\ &\quad - \frac{\Delta x_{\mathrm{W}} \Delta x_{\mathrm{E}}}{6} \frac{\partial^3 u}{\partial x^3} \\ \left(\frac{\mathrm{d}^2 u}{\mathrm{d}x^2}\right)_{\mathrm{P}} &= \left(\frac{u_{\mathrm{E}} - u_{\mathrm{P}}}{\Delta x_{\mathrm{E}}} - \frac{u_{\mathrm{P}} - u_{\mathrm{W}}}{\Delta x_{\mathrm{W}}} \right) \frac{2}{\Delta x_{\mathrm{E}} + \Delta x_{\mathrm{W}}} \\ &\quad + \frac{1}{3}\left(\Delta x_{\mathrm{E}} - \Delta x_{\mathrm{W}}\right) \frac{\partial^3 u}{\partial x^3} - \frac{\Delta x_{\mathrm{E}}^3 + \Delta x_{\mathrm{W}}^3}{12(\Delta x_{\mathrm{E}} + \Delta x_{\mathrm{W}})} \frac{\partial^4 u}{\partial x^4} \end{aligned} \tag{10.4}$$

where

$$\begin{aligned} x_{\mathrm{W}} &= x_{i-1} \\ x_{\mathrm{P}} &= x_i \\ x_{\mathrm{E}} &= x_{i+1} \\ \Delta x_{\mathrm{W}} &= x_{\mathrm{P}} - x_{\mathrm{W}} \\ \Delta x_{\mathrm{E}} &= x_{\mathrm{E}} - x_{\mathrm{p}} \end{aligned}$$

These expression are derived from Taylor-series and correspond to a local approximation of the field u using a second order polynomial basis. It is seen that while the first derivative has a truncation error of order 2 for arbitrary nodal spacings, the second derivative in general has a truncation error of order 1. Only in the special case of evenly distributed nodes, where $\Delta x_W = \Delta x_E$, a truncation error of order 2 is obtained for the second derivative.

Thus, since we do not have evenly distributed nodes in the simulation program, we cannot expect more that first order convergence of the diffusive term. Consequently the overall convergence rate becomes 1. This result is well supported by the experimentally found convergence of I_u. The measurement based on u_{center} is a little bit smaller, but this could also be related to the fact that the true value of u_{center} is not known, but only approximated (this makes the evaluation of the error less accurate).

In total we may conclude that the results from the lid driven cavity problem indicates first order convergence of the spacial discretization.

We end this section by examining the dependency of the time step on spacial resolution, and the time spent for the simulation as a function of the number of computational nodes. In figure 10.4 the time step Δt is plotted against the spacial resolution Δx (left part), and the average time spent per time step is plotted against the number of nodes (right part).

It is seen that the step size depends more or less linearly on the spacial resolution. This corresponds to a CFL-like condition

$$\mathrm{CFL} = \frac{\max(U + a)\Delta t}{\Delta x} \tag{10.5}$$

where

U speed of fluid particle
a speed of sound in fluid

The average time spent per time step is seen to depend approximately linearly on the total number of nodes. This observation is not a surprise, since we have used a fully explicit formulation. The cost for each node is constant, although, different from interior nodes and nodes on the boundary. The neighbor search algorithm also has complexity $O(N)$, where N is the total number of nodes (see section 9.3).

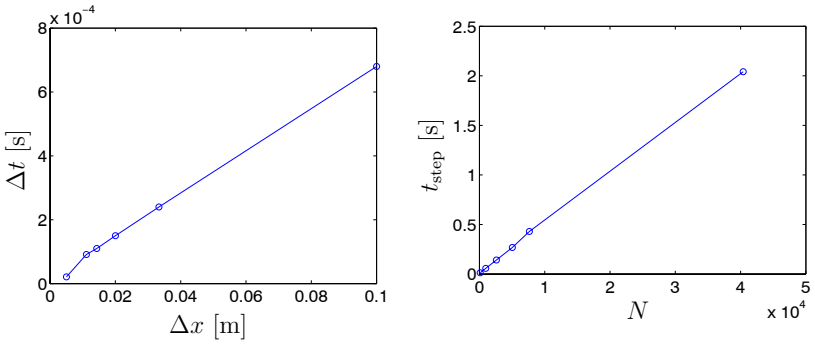

Figure 10.4: Left: The maximum allowed step size Δt as a function of the spacial resolution Δx. Right: Average time spent per time step (wall clock) as a function of the total number of nodes.

10.2 Fixed incline slider bearing

The fixed incline slider bearing is one of the standard cases solved using the Reynolds equation. In the usual setting the bearing consists of two planar surfaces. The upper part is fixed and inclined by some angle. The lower part is horizontal and moving at a constant speed in the horizontal direction. The inlet and outlet pressures are specified as boundary conditions.

We have solved the fixed incline slider problem using the following parameters

$$
\begin{aligned}
L &= 1 \quad \text{length of bearing, mm} \\
h_{\text{in}} &= 25 \quad \text{inlet height, } \mu\text{m} \\
h_{\text{out}} &= 15 \quad \text{outlet height, } \mu\text{m} \\
\mu &= 0.05 \quad \text{dynamic viscosity, Pa s} \\
\rho_0 &= 900 \quad \text{initial density, kg/m}^3 \\
p_0 &= 101500 \quad \text{reference pressure at } \rho = \rho_0\text{, Pa} \\
d_3 &= 10^6 \quad \text{parameter for Dowson-Higginson equation of state, Pa} \\
u_0 &= 1 \quad \text{velocity of lower part, m/s} \\
\text{time integrator} &\quad \text{Explicit 3rd order Runge-Kutta} \\
\text{time step} &\quad \text{Fixed} \\
\text{eos} &\quad \text{Dowson-Higginson} \\
\text{density bc} &\quad \text{1st order extrapolation}
\end{aligned}
\tag{10.6}
$$

It is noted that we have altered one parameter for the Dowson-Higginson equation of state (6.17) from $d_3 = 10^9$ Pa to $d_3 = 10^6$ Pa. This has been done in order to allow a bigger time step, thereby reducing the time required for carrying out

the simulations. With this change the speed of sound becomes of the magnitude $a = 45$ m/s and the Mach-number is then $Ma = 2.2 \cdot 10^{-2}$. The Reynolds number (based on the average film thickness) becomes $Re = 0.36$.

We apply the remeshing algorithm at every 10 time step (no matter the size of the time step). The remeshing is done soon enough to avoid nodes from moving beyond the outlet section. Because the fluid has the same dominant direction of flow throughout the domain, the general remeshing method cannot be used for long simulations (see section 9.8.3). Therefore a custom remeshing algorithm has been implemented of this test problem. Using a predefined number of nodes N along the lower part of the bearing we calculate the spacial resolution Δx as

$$\Delta x = \frac{L}{N-1} \tag{10.7}$$

We then distribute nodes at sections $x = i\Delta x$, $i = 1, 2, 3, \ldots$ and adjust the number of nodes in the y-direction so that the spacing Δx is achieved approximately.

Special treatment of the inlet and the outlet is also necessary, since we have not so far discussed boundary conditions of that kind. In accordance with the boundary conditions for the Reynolds equation, we enforce directly a prescribed density (pressure) at the nodes located on the inlet or outlet. At least two possibilities are available for the velocity components 1) extrapolation from the interior, 2) explicit calculation using the Reynolds equation. Extrapolation from the interior is a usual procedure for outlets, and it is adopted here. At the inlet it is common practice to specify all velocity components and density (pressure). We could actually compute the velocity components at the inlet using the Reynolds equation and the expressions (2.44) and (2.46). However, these results are valid for incompressible fluids only. Since we have "softened" the equation of state we will indeed experience effects of compressibility, so (2.44) and (2.46) would only be approximations of the true velocity distributions. Instead we choose to extrapolate the velocity components also at the inlet. It should be emphasized that the extrapolation of velocity at the inlet/outlet includes coupling with nodes on the solid boundaries. Without this coupling the extrapolation method is *not* stable.

In figure 10.5 we have plotted the evolution of the density field and the pressure evaluated at the lower part of the bearing, using $N = 300$ nodes along the lower part. The blue curve represents the steady state solution obtained from a solution of the compressible Reynolds equation, and the red curve denotes the pressure as obtained from the Navier–Stokes equations (the solutions of course share the same equation of state).

In table 10.2 we have shown the simulation results for various discretizations. The fourth column contains pressure as calculated by the compressible Reynolds equation integrated over the lower part I_{Reynolds}, using the trapezoidal rule (that is, the load carrying capacity of the bearing per unit width). Similarly the fifth column contains the same integral based on the Navier–Stokes solution. The last

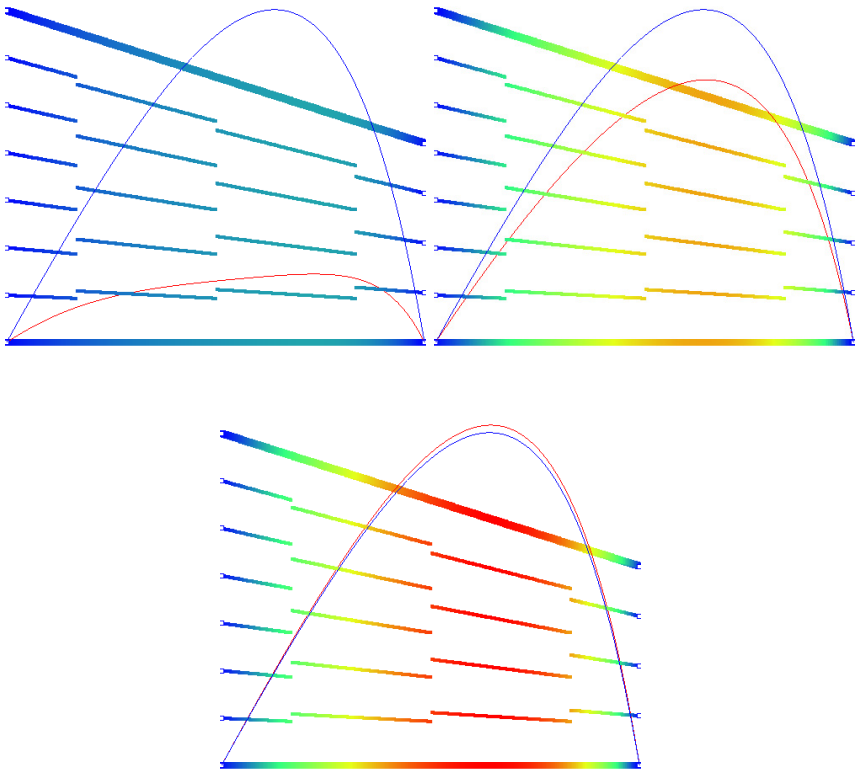

Figure 10.5: Evolution of the density field and pressure distribution of the fixed incline slider problem. Red color means high density, while blue color denotes low density. Blue pressure curve: Reynolds equation (steady state). Red pressure curve: Navier–Stokes equations. Note that the aspect ratio is not 1:1. Top Left: $t = 2.40 \cdot 10^{-7}$ s, Top Right: $t = 1.32 \cdot 10^{-6}$ s, Bottom: $t = 6.00 \cdot 10^{-6}$ s (converged).

N	Δx [m]	Δt [s]	I_{Reynolds} [N/m]	$I_{\text{Navier-Stokes}}$ [N/m]	T_{elapsed} [s]
200	5.03E-6	2.0E-7	1.3387938E+2	1.41434E+2	215.6
300	3.34E-6	1.2E-7	1.3387996E+2	1.34612E+2	804.5
400	2.51E-6	7.4E-8	1.3388016E+2	1.34215E+2	2478.7
500	2.00E-6	4.6E-8	1.3388025E+2	1.34059E+2	5135.2
600	1.67E-6	3.1E-8	1.3388030E+2	1.33990E+2	11069.7
800	1.25E-6	1.7E-8	1.3388035E+2	1.33936E+2	–
1000	1.00E-6	1.1E-8	1.3388037E+2	1.33897E+2	–
1500	6.67E-7	5.0E-9	1.3388040E+2	1.33901E+2	–

Table 10.2: Simulation results from the fixed incline slider problem.

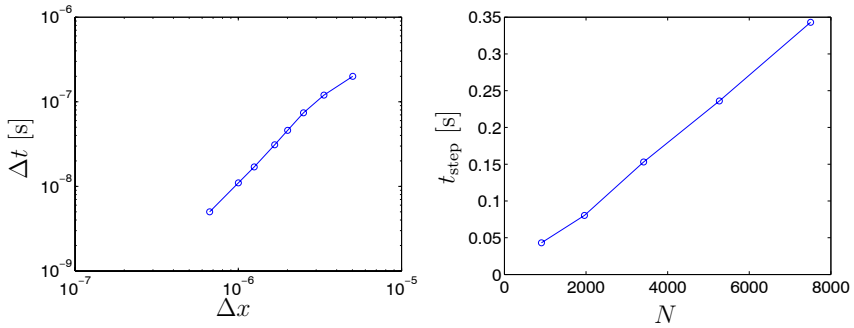

Figure 10.6: Left: Time step Δt as a function of mean discretization length Δx. Right: Average time spent per time step as a function of the total number of nodes.

column holds the total time elapsed for performing the simulations. The missing values are due to wrapping of the timer-function used in the simulation program. However, for $N = 1500$ the simulation time was about 4,5 days.

In figure 10.6 (left part) we have plotted the maximum allowed step size Δt as a function of the spacial resolution Δx. If we exclude the data point corresponding to the most coarse resolution we may compute the slope of the regression line as $\alpha = 2.01$. This indicates a Fourier-like limit on the step size

$$Fo = \frac{C\mu\Delta t}{(\Delta x)^2} \tag{10.8}$$

where C is some constant.

In figure 10.6 (right part) we have plotted the average time spent for one time step as a function of the total number of nodes. A linear relationship is observed, as was also the case for the lid driven cavity problem.

In figure 10.7 we have shown the pressure distribution using $N = 1500$. Three solutions obtained from the incompressible Reynolds, the compressible Reynolds,

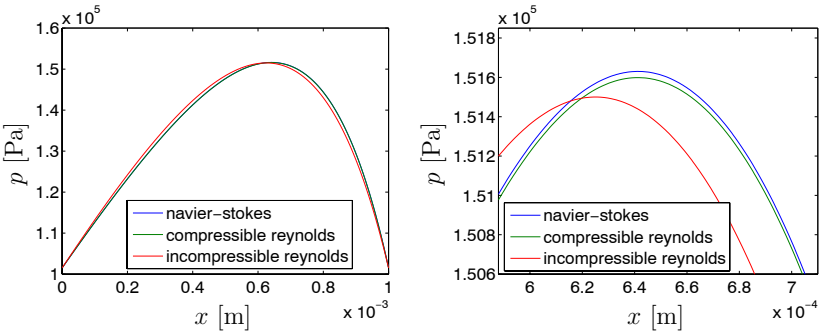

Figure 10.7: Left: Plots of the pressure distribution as calculated from 1) incompressible Reynolds, 2) compressible Reynolds, 3) compressible Navier–Stokes. Right: The same thing with zooming on the pressure peak.

and Navier–Stokes equations, respectively, are plotted. It is seen that the effect of compressibility results in a slightly higher maximum pressure, and the pressure peak is shifted downstream. Also a small difference between the Reynolds solution and the Navier–Stokes solution can be observed.

In order the asses the convergence properties for the fixed incline slider bearing we define the following measure of the error

$$E = |I^N_{\text{Navier–Stokes}} - I^{1500}_{\text{Navier–Stokes}}| \tag{10.9}$$

which may be evaluated for $N = 200, 300, 400, 500, 600, 800, 1000$. In figure 10.8 we have plotted E against Δx. If cases $N = 200$ and $N = 1000$ are neglected we may calculate the slope of the linear part as $\alpha = 3.083$. This result is unexpected, as compared to the lid driven cavity, for which we found the spacial discretization to converge by order 1. There is, however, a good explanation for this behavior. In figure 10.9 we have plotted the volume flow rate against Δx, by integrating u-velocity from $y = 0$ to $y = h_{\text{in}}$ at $x = 0$. From the compressible Reynolds solution with $N = 1500$ the volume flow rate becomes (2.42)

$$q^{\text{Reynolds}}_{\text{in}} = 9.55 \cdot 10^{-6} \ \ \text{m}^3/\text{m} \tag{10.10}$$

Clearly, the Navier–Stokes solution is not subjected to the correct boundary condition at the inlet. For the highest resolution $N = 1500$ we have

$$q^{\text{Navier-Stokes}}_{\text{in}} = 8.89 \cdot 10^{-6} \ \ \text{m}^3/\text{m} \tag{10.11}$$

which is quite close but not really the right value. From figure 10.9 it is seen that the situation quickly becomes worse as resolution is decreased. This is the reason for the seemingly very fast convergence rate of 3 order – the boundary conditions are highly affected by the resolution, which again influences the pressure distribution. Therefore we conclude that we cannot measure the convergence rate of the

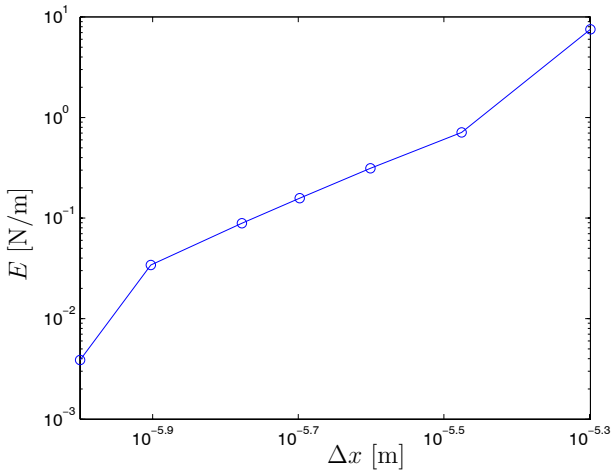

Figure 10.8: Error as defined in (10.9) as a function of the discretization length Δx.

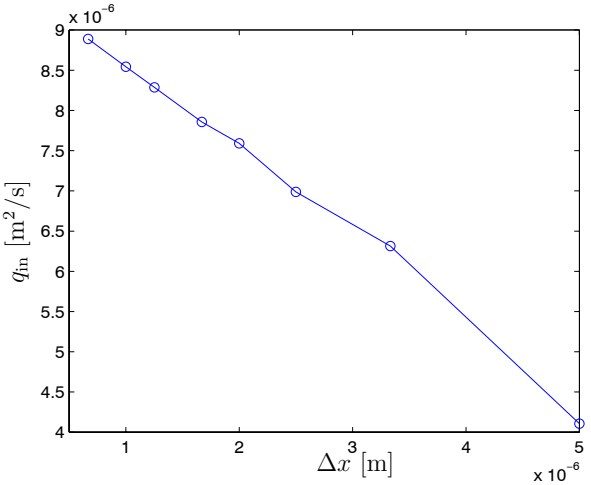

Figure 10.9: The volume flow rate per unit width as a function of the discretization Δx.

spacial discretization using this test case. However, we have demonstrated that the pressure distribution obtained from the Navier–Stokes solution converges to the compressible Reynolds solution.

10.3 Rotating free surface

In this section we investigate a problem for the free surface boundary condition. The simulation problem consists of a circular shaft, rotating at constant speed, which is fully immersed in fluid. The fluid is shaped as an annulus with boundaries consisting of the shaft and a free surface. These are the simulation parameters

$$
\begin{aligned}
r_{\text{shaft}} &= 5 \quad \text{radius of shaft, mm} \\
r_{\text{free}} &= 12 \quad \text{radius at free surface boundary, mm} \\
\omega_0 &\quad \text{rotation speed of shaft, rad/s} \\
\rho_0 &= 900 \quad \text{initial density, kg/m}^3 \\
\mu_0 &= 10 \quad \text{dynamic viscosity, Pa s} \\
\sigma_0 &= 0 \quad \text{surface tension, N/m} \\
p_0 &= 101500 \quad \text{reference pressure at } \rho = \rho_0 \\
d_3 &= 10^6 \quad \text{parameter for Dowson-Higginson equation of state, Pa} \\
\text{time integrator} &\quad \text{Explicit or implicit 3rd Runge-Kutta} \\
\text{time step} &\quad \text{Fixed} \\
\text{eos} &\quad \text{Dowson-Higginson} \\
\text{density bc} &\quad \text{Neumann condition}
\end{aligned}
\tag{10.12}
$$

As in the previous section we have altered the Dowson-Higginson equation of state in order to allow bigger time steps (and less simulation time required). The Reynolds number ranges from $Re \approx 0.3$ to $Re \approx 50$, depending on the rotation speed of the shaft.

The remeshing algorithm is engaged at every 20 time step, no matter the size of the time step. For this problem we employ the general remeshing method described in section 9.8. The initial nodal layout is generated by specifying the number of nodes along a radius. From this the nodal spacing Δx is calculated. At a given radius nodes are distributed in the circumferential direction such that a spacing of Δx in that direction is achieved approximately.

This simulation problem is a test for the free surface boundary condition. After the initial transient the fluid should rotate as a rigid body with the same angular velocity as the shaft – since the shear stress on the free surface should be zero. This property can be tested by comparing the speed at a given radius with the exact value for a rigid body. In figure 10.10 we have plotted the transient

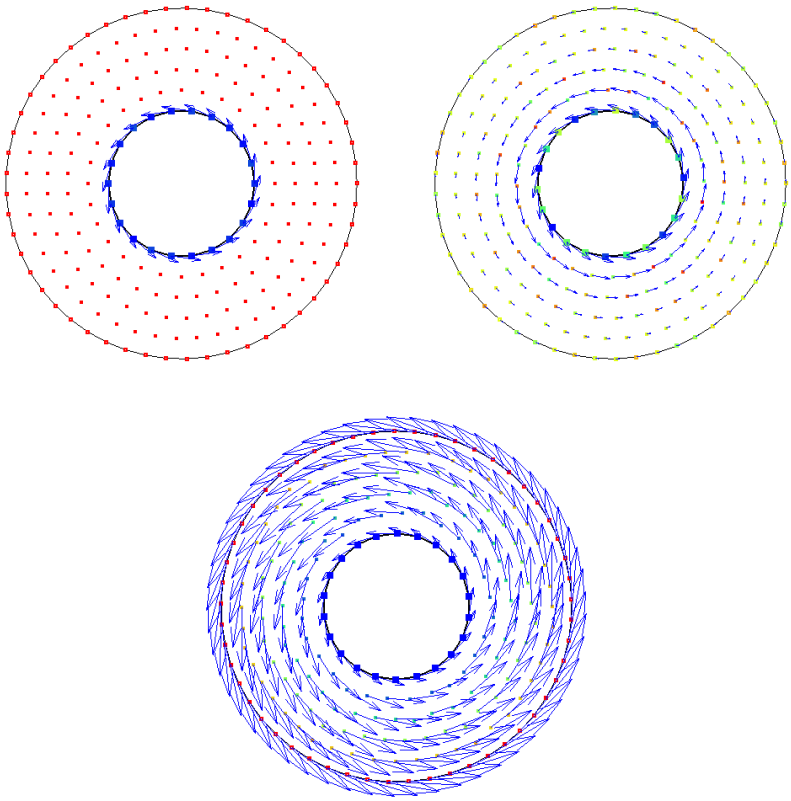

Figure 10.10: Evolution of the rotating free surface test problem. Note that the color palette for density is adjusted on the fly, so that the full color range is used at all times. Top left: $t = 0$ s, top right: $t = 7.8 \cdot 10^{-5}$ s, bottom: $t = 5 \cdot 10^{-2}$ s (final time).

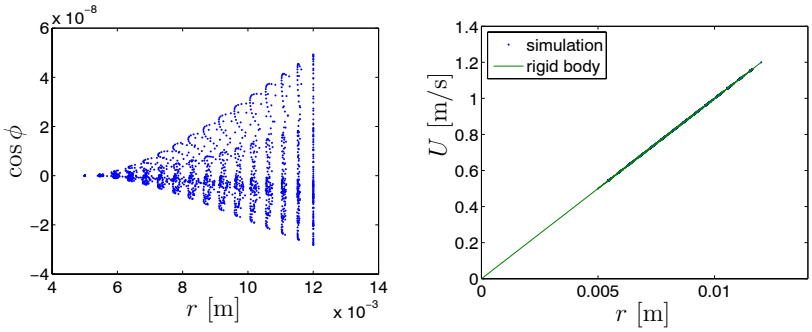

Figure 10.11: Left: Dot product of position vector (x_i, y_i) and velocity vector (u_i, v_i). Right: Speed.

evolution of the system at three moments in time. Initially the density is even everywhere. After the transient, however, density increases with increasing radius due to the centrifugal force. In figure 10.11 we have plotted values taken after the transient phase. The figure shows the speed

$$U_i = \sqrt{(u_i)^2 + (v_i)^2} \tag{10.13}$$

evaluated at every node i, as a function of the radius

$$R_i = \sqrt{(x_i)^2 + (y_i)^2} \tag{10.14}$$

For comparison the speed for a rigid body (given by $U = R\omega$) is also shown. It is seen that agreement is excellent. The left part of figure 10.11 shows a plot of the dot product between the position vector (x_i, y_i) and the velocity vector (u_i, v_i) as a function of radius R

$$\cos\phi_i = x_i u_i + y_i v_i \tag{10.15}$$

where ϕ_i is the angle between the position vector and the velocity vector at node i. The dot product should always be zero, since the velocity is perpendicular to the radius on a rotating rigid disc. It is seen that the maximum error is of the magnitude 10^{-8}, which is well within reasonable tolerance. We may therefore conclude that the condition of zero shear stress on the free surface is implemented correctly.

So far we have not discussed the difference between the explicit time integrator and the implicit one. However, now is a good time to investigate the maximum allowed step size of each method. A constant angular velocity of $\omega = 100$ rad/s is used in the following. The results for different discretizations are shown in table 10.3, and graphically in figure 10.12.

The implicit time integrator needs to solve nonlinear systems of equations in every time step, which is done using the simplified Newton-Raphson method. For

N	$\Delta t_{\text{explicit}}$ [s]	$\Delta t_{\text{implicit}}$ [s]	$t_{\text{elapsed}}^{\text{explicit}}$ [s]	$t_{\text{elapsed}}^{\text{implicit}}$ [s]
5	7.8E-5	1.6E-3	13.08	23.09
10	3.0E-5	5.8E-4	88.94	286.19
15	1.2E-5	3.1E-4	426.81	1735.16
20	7.1E-6	2.1E-4	1179.03	4410.20
25	4.5E-6	1.7E-4	2790.75	13249.50

Table 10.3: Simulation results from the rotating disc problem.

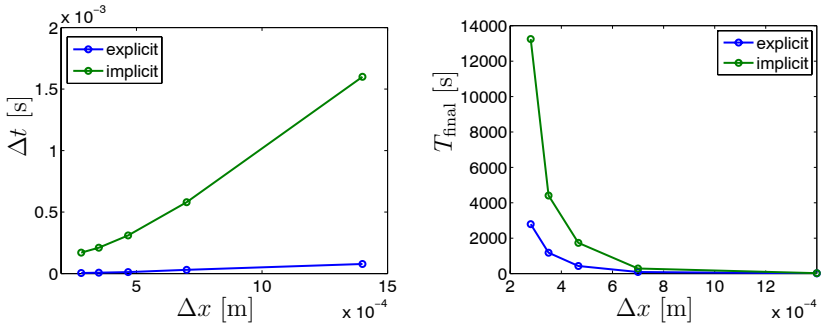

Figure 10.12: left: Maximum allowed time step. Right: Total time elapsed for simulation.

this particular test problem a new Jacobian matrix is generated at every 20 time step (right after the remeshing method, which might alter the number of nodes). Typically, each stage needs 4 to 6 iterations. Since the method has three implicit stages, it will evaluate the right hand side roughly 15 times. The explicit time integrator on the other hand needs 3 right hand sides for one time step. Thus, the implicit time step must be *at least* 5 times greater than the explicit one to be competitive, with respect to simulation time. Extra work for matrix factorization and backsubstitution is needed for the implicit method, so in practice the implicit time step must be even greater to get a saving in computation time. It should be noted that the update rate of the Jacobian matrix is problem-dependent.

From the table or the graphics it is seen that the implicit time integrator allows much bigger time steps than the explicit method. The ratio between implicit and explicit time steps is about 20–38, the high value obtained for the highest spacial resolution. This indicates that the implicit method becomes more attractive as spacial resolution is increased. Unfortunately the scaling with the number of nodes is not favorable for the implicit method. In figure 10.13 we have shown the average time spent per time step for the implicit and the explicit time integrator. For the explicit method we observe once more a linear relationship, as in the two previous sections. For the implicit method, however, the scaling with the number of nodes is *progressive*. This happens because of the needed matrix factorizations,

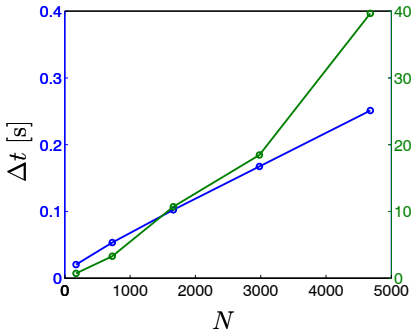

Figure 10.13: Average time spent for one time step of the free surface problem 10.12 as a function of the total number of nodes N. Blue: Explicit method. Green: Implicit method.

which in general scale as $O(N^3)$ with N being the size of the system.

We may therefore conclude that the implicit method is a good choice for high spacial resolution and a limited number of nodes. Unfortunately, the typical situation is that fine spacial resolution induces a large number of nodes. This effect is seen in figure 10.12 where the total computation time is plotted against spacial resolution. For the problems we have dealt with in this work we always found the explicit method to be the fastest.

10.4 Drop oscillation

In this section we examine the free surface boundary condition further by simulating the oscillations of a drop of liquid. The initial configuration is a circular drop which has been perturbed into an ellipse. When the simulation begins the drop will deform due to the action of the surface tension. Here are the simulation parameters

$$\begin{aligned} L_x &= 1.000 \quad x\text{-semi axis, m} \\ L_y &= 1.005 \quad y\text{-semi axis, m} \\ \rho_0 &= 1 \quad \text{initial density, kg/m}^3 \\ \mu_0 &= 10^{-6} \quad \text{dynamic viscosity, Pa s} \\ \sigma_0 &= 1 \quad \text{surface tension, N/m} \\ \Delta t &= 5 \cdot 10^{-4} \quad \text{fixed time step, s} \\ \text{time integrator} &\quad \text{Explicit 3rd Runge-Kutta} \\ \text{density bc} &\quad \text{Neumann condition} \\ \text{eos} &\quad \text{Ideal gas} \end{aligned} \tag{10.16}$$

The motion of drops in general is very complex. However, when the perturbation from equilibrium (a circle) is small it makes sense to speak of *oscillations* having a well-defined amplitude and frequency. Analytical solutions are available for special limiting cases as well as higher order analyses with a wider range of validity. Lord Rayleigh [49] has derived an expression for the frequency for the case of small perturbations and zero viscosity

$$\omega_n^2 = \frac{\sigma_0}{\rho_0 R^3} n(n^2 - 1) \tag{10.17}$$

where $n = 1$ corresponds to rigid body motion and $n = 2$ corresponds to the first mode of oscillation. This formula is derived for a two dimensional drop (another formula also derived in [49] exists for the 3D case).

In figure 10.14 we have plotted the initial configuration and two later time steps for the drop under consideration. As can be seen the fluid is initially at rest, except for nodes on the boundary. Imposition of the boundary conditions on the free surface results in non-zero velocity components due to the surface tension. Because the fluid is compressible the velocity distribution due to curvature is superposed with a (small) net contraction of the drop.

The simulation is performed with a fixed time step, and the remeshing algorithm is engaged at every 100 time step. The gas constant for the ideal gas law is taken as $C_0 = 2500$ J/kg, corresponding to a speed of sound $a = 50$ m/s. Instead of eliminating viscosity completely, we use a small value of $\mu = 10^{-6}$ Pa s. This is necessary because the free surface boundary condition implemented in the simulation program is not valid for $\mu = 0$. In fact, in (8.8, 2nd equation) we have eliminated μ by division, which is only valid for non-zero μ. If viscosity is zero the shear stress condition is satisfied *always*. In that case (8.8, 2nd equation) must be replaced by the requirement that the velocity vector is always perpendicular to the free surface. That condition is not implemented, so to be correct we use $\mu = 10^{-6}$ Pa s as an approximation of zero viscosity.

We measure the drop motion by calculating the ratio of the main axes of the ellipsoid throughout the simulation

$$A = \frac{L_x}{L_y} \tag{10.18}$$

In figure 10.15 we have plotted the transient response. The detection of wave tops is also shown. From this information the period can be calculated by

$$T = \frac{t_N - t_1}{N - 1} = \frac{9.0550 \text{ s} - 1.2950 \text{ s}}{4 - 1} = 2.5867 \text{ s} \tag{10.19}$$

where N is the number of peaks. The frequency as obtained by the simulation program and using the analytical formula (10.17) with $n = 2$ then becomes, respectively

$$\begin{aligned} \omega_{\text{numerical}} &= \frac{2\pi}{T} = 2.4291 \text{ rad/s} \\ \omega_{\text{analytical}} &= \sqrt{6} = 2.4495 \text{ rad/s} \end{aligned} \tag{10.20}$$

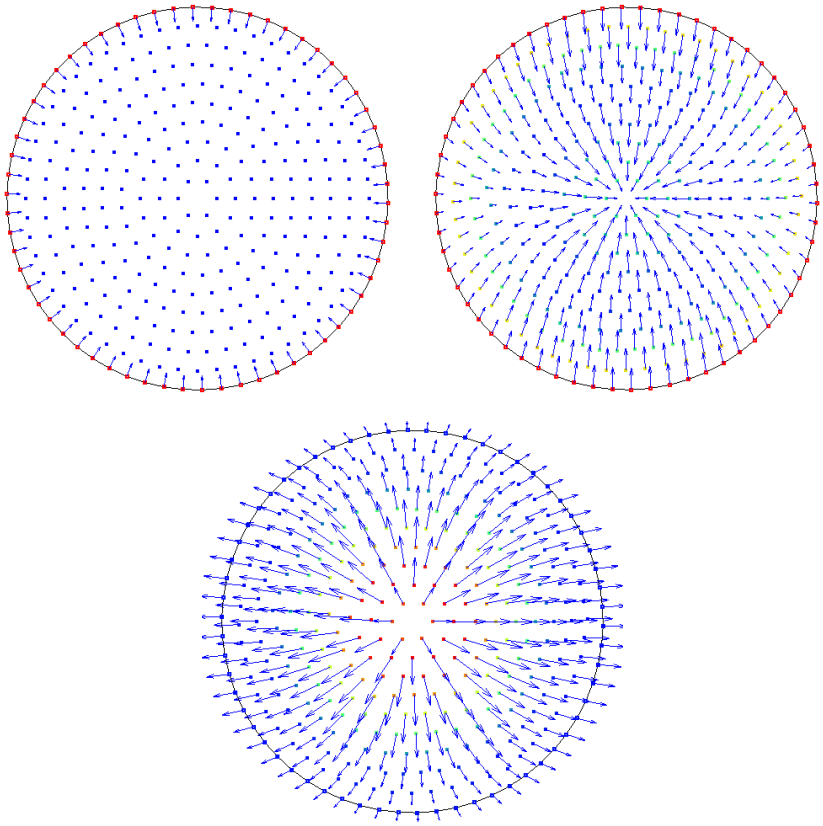

Figure 10.14: The oscillating drop test problem. Top left: $t = 0$ s, top right: $t = 2.75 \cdot 10^{-2}$ s, bottom $t = 3.05 \cdot 10^{-2}$ s.

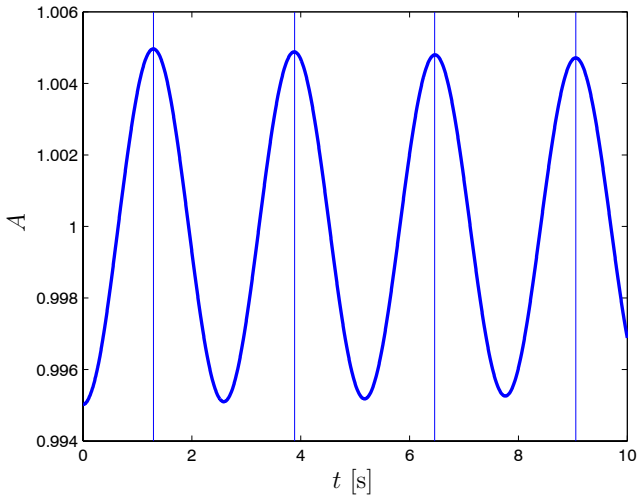

Figure 10.15: The ratio between the main axis of the oscillating drop plotted against time.

The agreement is seen to be very good, the relative deviation being approximately 0.8%. This test problem indicates that the free surface boundary condition (including the surface tension term) is implemented correctly.

10.5 Piston ring simulation I

In this section we carry out a simulation including a single piston ring. The main simulation parameters are given below

$$
\begin{aligned}
b &= 2 \;\; \text{width of ring, mm} \\
s_h &= 100 \;\; \text{barrel height of ring, } \mu\text{m} \\
r &= 200 \;\; \text{radius of arcs, } \mu\text{m} \\
h_{\min} &= 90 \;\; \text{minimum film thickness under piston ring, } \mu\text{m} \\
h_\infty &= 60 \;\; \text{undisturbed film thickness on liner, } \mu\text{m} \\
x_{\text{trp}} &= \pm 730 \;\; \text{initial pos. of triple points rel. to ring center, } \mu\text{m} \\
O &= 10000 \;\; \text{crank rotations per minute, rpm} \\
s &= 5 \;\; \text{stroke, mm} \\
\rho_0 &= 870 \;\; \text{initial density, kg/m}^3 \\
p_0 &= 100500 \;\; \text{reference pressure at } \rho = \rho_0, \text{ Pa} \\
\mu_0 &= 0.05 \;\; \text{dynamic viscosity, Pa s} \\
\sigma_0 &\;\; \text{surface tension (to be defined), N/m} \\
d_3 &= 10^7 \;\; \text{parameter for Dowson-Higginson formula, Pa} \\
\text{time integrator} &\;\; \text{Explicit 3rd Runge-Kutta} \\
\text{density bc} &\;\; \text{Neumann condition} \\
\text{eos} &\;\; \text{Dowson-Higginson-Wallis} \\
\text{triple point model} &\;\; \text{dynamic contact angle} \\
\phi_{\text{ss}} &= \frac{\pi}{2} \;\; \text{steady state contact angle} \\
C_0 &= 3 \;\; \text{tuning parameter for dynamic contact angle}
\end{aligned}
\tag{10.21}
$$

A few comments about the parameters and limitations are appropriate. Please note, that we postpone the discussion of possible remedies for the limitations until section 10.7.

First we explain the shape of the piston ring. The main geometry is given by a parabola, defined such that $y(0) = 0$ and $y(\pm b/2) = s_h$. Instead of just attaching vertical lines at $x = \pm b/2$ we have inserted circular arcs to make the transition smooth. The arcs are matched so that the tangent is continuous along the piston ring surface.

The minimum film thickness under the piston ring is fixed at $h_{\min} = 90$ μm. This value is somewhat too big, since piston rings in reality experience from 0 to about 20 μm separation from the liner. However, we use a relatively thick oil film in order to be able to carry out the simulation using a reasonable amount of time. Also, the value is not constant but varies depending on the loading of the

ring and the amount of oil under the ring. The lateral movement of the piston ring is not considered in this work.

The horizontal movement is taken to be given by

$$x = \frac{s}{2}\cos(\frac{2\pi}{60} \cdot O \cdot t) \tag{10.22}$$

Of course a stroke of 5 mm and 10000 revolutions per minute does not correspond to a large Diesel engine. However, by limiting the excursion of the piston we can save discretization nodes along the liner. If the discretization length is $\Delta x = 2$ μm, oil film thickness is 8 μm, and the stroke is 2,5 m (these are realistic figures for a big engine) we would need roughly 5,000,000 nodes. With the current setup we only need about 6,400 nodes. Selecting a high number of revolutions reduce the period making it $T = 60$ msec instead of about one second for large Diesel engines. The maximum piston speed obtained with our settings is 2.62 m/s, while the real life value is about 15 m/s.

The discretization parameters are

$$\begin{aligned}
\Delta t &= 5 \cdot 10^{-7} \quad \text{fixed time step, s} \\
\Delta x &= 1.2 \cdot 10^{-5} \quad \text{average spacial discretization of fluid, m} \\
h_{\text{fluid}} &= 3.24 \cdot 10^{-5} \quad \text{smoothing length at interior nodes, m} \\
h_{\text{solid}} &= 5.76 \cdot 10^{-5} \quad \text{smoothing length at solid boundaries, m} \\
h_{\text{freesurface}} &= 5.76 \cdot 10^{-5} \quad \text{smoothing length at free surfaces, m} \\
h_{\text{triplepoint}} &= 6.6 \cdot 10^{-5} \quad \text{smoothing length at triple points, m}
\end{aligned} \tag{10.23}$$

Remeshing is engaged at every 10 time step. For this simulation we use the general remeshing algorithm as described in section 9.8.3.

In figure 10.16 we have shown snap shots from the simulation of (10.21) at four different instants of time covering almost one period. The global view of system is given with aspect ratio 1:4 (the y-direction is magnified four times). Note that the color palette for density is scaled on the fly so that the full range of colors is used at all times. The surface tension was set to $\sigma = 0$ N/m for this simulation.

In figure 10.17 we have repeated the same simulation, this time with surface tension set to $\sigma = 0.07$ N/m. The view is centered around the piston ring and the aspect ratio is 1:1. The effect of surface tension is easily seen, since in this case curvature of the free surface is reduced.

Finally, we plot the position of the left triple point on the piston ring in figure 10.18. The data is collected over 5 revolutions. It can be seen that the motion of the triple point is quasi-periodic.

From the results of this section we may conclude that the simulation program is in fact able to perform calculations on piston ring lubrication. The main problem, which must be addressed is the time required for a single simulation. For one revolution of the problem given by (5.18) the time spent on a 3 GHz

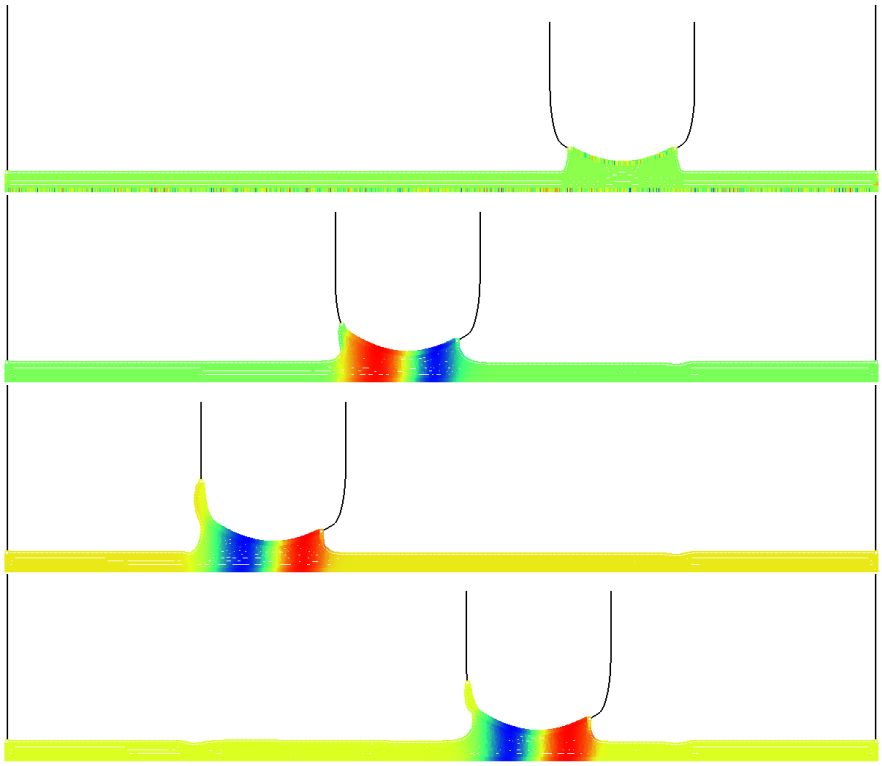

Figure 10.16: Simulation results for (5.18). The surface tension is set to $\sigma = 0$ N/m. Top: $t = 2.0 \cdot 10^{-5}$ s, upper middle: $t = 1.7 \cdot 10^{-3}$ s, lower middle: $t = 3.4 \cdot 10^{-3}$ s, bottom $t = 5.1 \cdot 10^{-3}$ s.

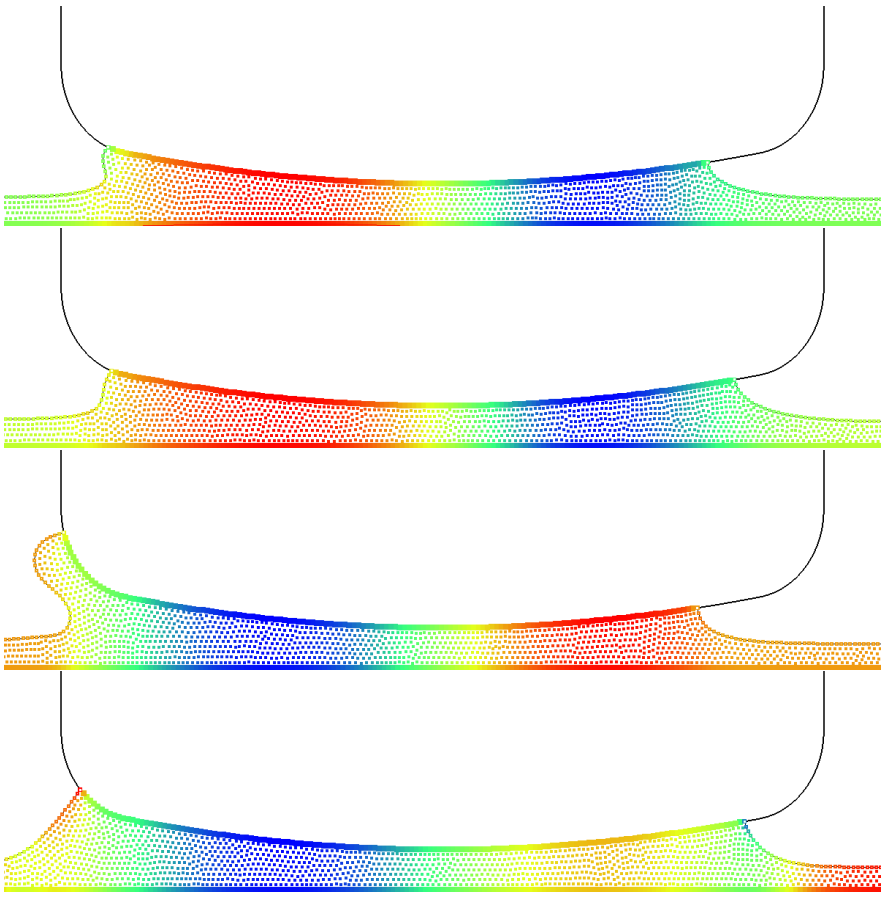

Figure 10.17: Comparison of results with surface tension and without surface tension. Top: $t = 1.5 \cdot 10^{-3}$ s, $\sigma = 0$ N/m. Upper middle: $t = 1.5 \cdot 10^{-3}$ s, $\sigma = 0.07$ N/m. Lower middle: $t = 3.0$ s, $\sigma = 0$ N/m. Bottom: $t = 3.0$ s, $\sigma = 0.07$ N/m.

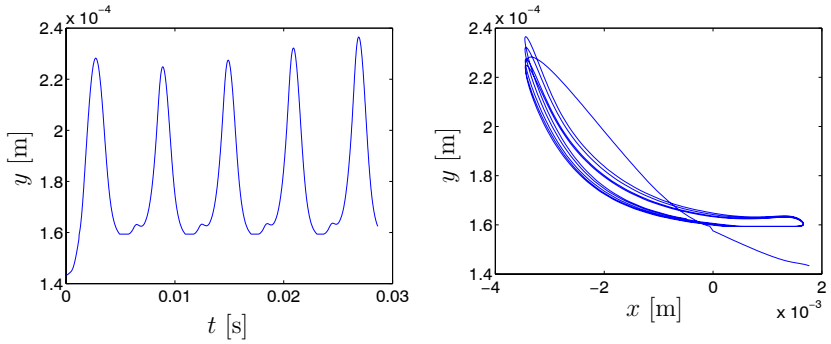

Figure 10.18: Location of the left triple point on the piston ring ($\sigma = 0.07$ N/m). Left: y-position against time. Right: y-position against x.

desktop computer is 1 h 11 min (or approximately 4300 seconds). This is a very reasonable cost *if* we were actually solving a realistic problem. However, the oil film thickness is exaggerated (90 μm instead of 9 μm) and the period is reduced significantly (from 1 s to 60 msec). In the most favorable case the time step scales linearly with the spacial discretization. Thus, if we assume that the number of nodes can be maintained (e.g. by "freezing" some parts of the system) the time required for one revolution of a realistic problem becomes

$$T = \frac{90\ \mu\text{m}}{9\ \mu\text{m}} \cdot \frac{1000\ \text{msec}}{60\ \text{msec}} \cdot 4300\ \text{s} \approx 200\ \text{h} \tag{10.24}$$

This corresponds to roughly eight days, and is even an optimistic estimate (the number of nodes could increase, and the time step might not scale linearly, see section 10.2). As an attempt to save nodes a combination of the Navier–Stokes equations and the Reynolds equation has been examined. This is the subject of the next section.

10.6 Piston ring simulation II

In this section we examine the possibility of combining the Navier–Stokes equations and the Reynolds equation in a single formulation. The idea is to employ the Reynolds equation on the part of the oil film under the piston ring. The Navier–Stokes equations will be utilized on the part with the free surface. An overlap between the to domains is established at the inlet (or outlet) of the piston ring. This is indicated in figure 10.19.

The idea is to calculate the pressure distribution under the piston ring using the Reynolds equation. The inlet point x_{in} is taken to coincide with the x-position of the relevant triple point x_{trp} at all times. The compressible Reynolds equation is solved subject to the Reynolds cavitation criteria. Once the pressure

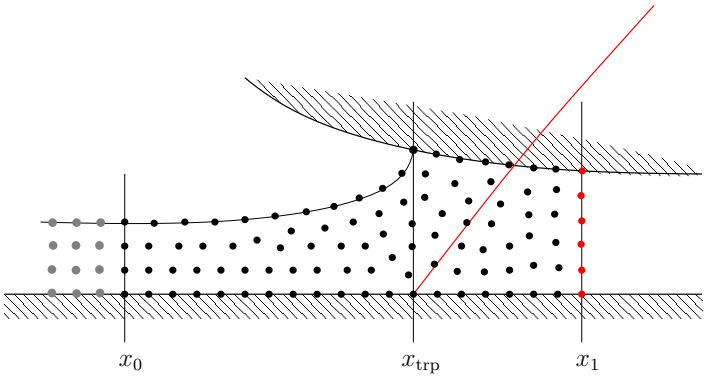

Figure 10.19: A sketch showing the overlap between the domains of the Navier-Stokes equations and the Reynolds equation. Gray nodes are "frozen" which means that they are not updated by the time integration. Black nodes are governed by the Navier–Stokes equations. For the red nodes density is specified using the value obtained by the Reynolds equation, and velocity components are extrapolated from black nodes.

distribution is found the density is also known. Thus we can enforce density at the "outlet" x_1 of the domain for the Navier–Stokes equations. The velocity components at x_1 are obtained by extrapolation from the remaining fluid nodes and the solid boundaries.

The method is tested for the following realistic problem

$$\begin{aligned}
b &= 20 \ \text{width of ring, mm} \\
s_h &= 100 \ \text{barrel height of ring, } \mu\text{m} \\
r &= 200 \ \text{radius of arcs, } \mu\text{m} \\
h_{\min} &= 10 \ \text{minimum film thickness under piston ring, } \mu\text{m} \\
h_\infty &= 8 \ \text{undisturbed film thickness on liner, } \mu\text{m} \\
x_{\text{trp}} &= \pm 7300 \ \text{initial pos. of triple points rel. to ring center, } \mu\text{m} \\
u_0 &= 10 \ \text{velocity of liner, m/s} \\
\rho_0 &= 870 \ \text{initial density, kg/m}^3 \\
p_0 &= 100500 \ \text{reference pressure at } \rho = \rho_0\text{, Pa} \\
\mu_0 &= 0.05 \ \text{dynamic viscosity, Pa s} \\
\sigma_0 &= 0.07 \ \text{surface tension, N/m} \\
d_3 &= 10^7 \ \text{parameter for Dowson–Higginson formula, Pa} \\
\text{time integrator} &\quad \text{Explicit 3rd Runge–Kutta} \\
\text{density bc} &\quad \text{Neumann condition} \\
\text{eos} &\quad \text{Dowson–Higginson} \\
\text{triple point model} &\quad \text{dynamic contact angle} \\
\phi_{\text{ss}} &= \frac{\pi}{2} \ \text{steady state contact angle} \\
C_0 &= 3 \ \text{tuning parameter for dynamic contact angle}
\end{aligned} \tag{10.25}$$

For this simulation problem we have taken the piston ring to be stationary and the liner to be moving horizontally. The target velocity is achieved by a smooth ramping up from $u_0 = 0$ m/s to $u_0 = 10$ m/s over 100 time steps (a cubic spline takes care of this). In figure 10.20 we have shown a plot of the piston ring in aspect ratio 1:1. The figure also shows (in blue) the part, which is discretized using the Navier-Stokes equations.

The discretization parameters are given by

$$\begin{aligned}
\Delta t &= 2 \cdot 10^{-8} \ \text{fixed time step, s} \\
\Delta x &= 2 \cdot 10^{-6} \ \text{spacial discretization of fluid, m} \\
h_{\text{fluid}} &= 5.4 \cdot 10^{-6} \ \text{smoothing length at interior nodes, m} \\
h_{\text{solid}} &= 9 \cdot 10^{-6} \ \text{smoothing length at solid boundaries, m} \\
h_{\text{freesurface}} &= 9 \cdot 10^{-6} \ \text{smoothing length at free surfaces, m} \\
h_{\text{triplepoint}} &= 1.1 \cdot 10^{-5} \ \text{smoothing length at triple points, m}
\end{aligned} \tag{10.26}$$

Remeshing is performed at every 5 time steps, in order to avoid nodes from crossing the line $x = x_1$ (the "outlet" of the Navier–Stokes domain). The total number of nodes is approximately 2300, and the final time $t = 5.02 \cdot 10^{-6}$ is

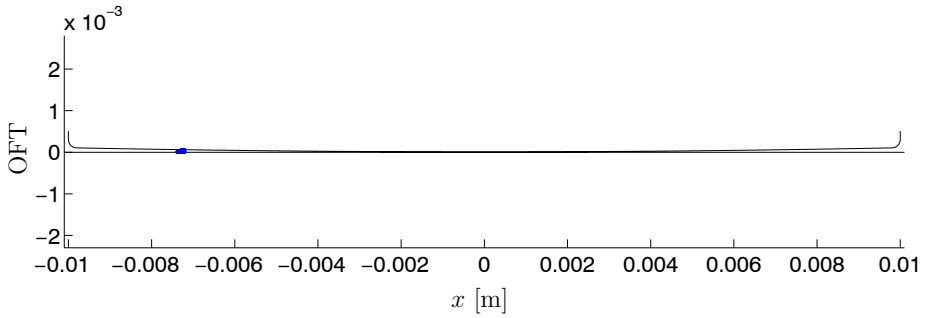

Figure 10.20: A plot of the configuration for simulation problem (10.25). Note, that only a small region is modeled using the Navier-Stokes equations (in blue). Aspect ratio 1:1.

reached in 47 seconds. In figure 10.21 we have shown the results of the simulation for two moments of time. The red curve denotes the pressure obtained from the Navier–Stokes solution evaluated on the liner, and the blue curve is the pressure distribution obtained from the Reynolds equation. At the beginning of the simulation pressure is high at the left end of the liner. This happens because of the vertical part, which is added to the liner, which acts as wave generator. We have included this part, in order to make the oil on the liner move as a rigid body. At $t = 4.4 \cdot 10^{-6}$ s the liner is already at the target velocity. A small bump on the free surface can be observed – this comes from the wave generator. The pressure curve shows that a pressure peak exists at the transition from the undisturbed oil film to the part where build up of oil takes place. In fact it is this increase of pressure that initiates the oil build up.

Looking at the region under the piston ring we see quite good agreement between the pressure from the Navier–Stokes solution and the Reynolds solution. This is of partly due to the fact, that density at x_1 is enforced to match the density obtained using the Reynolds equation.

At this point we have not examined how big the overlap between the Reynolds domain and the Navier–Stokes domain should be. For the current test case, it seems that the overlap could be made smaller. Also an obvious idea for saving more nodes, would be to introduce different discretization length in the x-direction and the y-direction. Finally, it should be mentioned that the Reynolds solution does not include inertia effects. This means that the pressure distribution is always fully developed. For slowly changing running conditions this is not a problem in relation to the pressure distribution obtained by the Navier–Stokes equations. However, if the quasi–static assumption does not hold it might be more difficult to mix the Reynolds equation and the Navier–Stokes equations in the way proposed here.

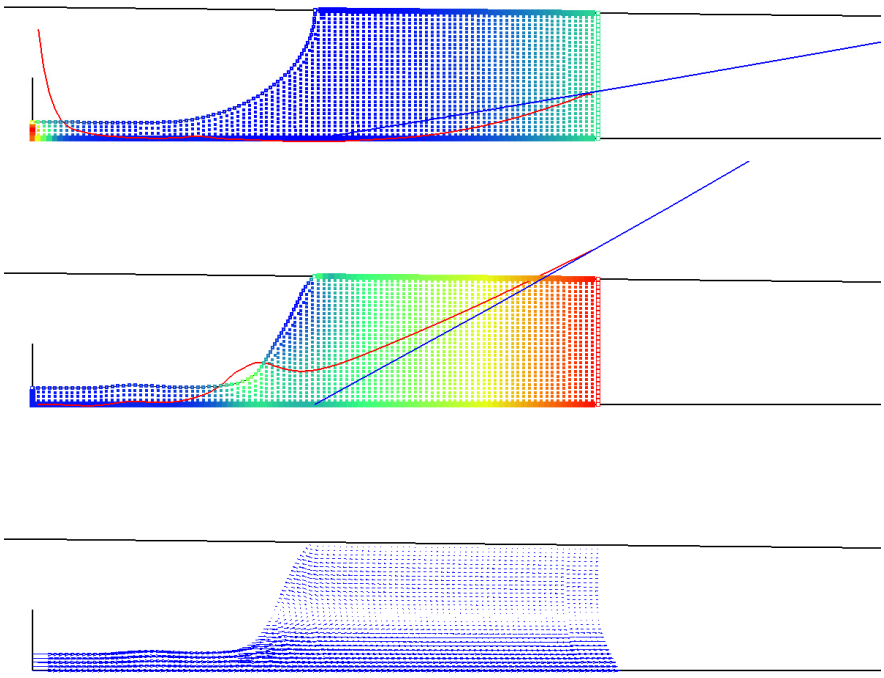

Figure 10.21: Simulation results for (10.25). Top: Density at $t = 7.6 \cdot 10^{-7}$ s, middle: Density at $t = 4.4 \cdot 10^{-6}$ s, bottom: velocity at $t = 4.4 \cdot 10^{-6}$ s.

10.7 Discussion and future work

In this section we sum up the findings related to the solution of the Navier–Stokes equations. Some of the results to appear below are not documented in the main text. However, it seems valuable to include the findings anyway.

Boundary condition for density Three different kinds of boundary conditions for density (pressure) have been considered in this work. The main trend from the test problems is that the Neumann boundary condition derived from the momentum equations is more stable than the extrapolation methods. This is especially true in relation to free surfaces, but also observed in test cases with only solid boundaries. The Neumann condition allows in general larger time steps than the first order extrapolation, which is in turn more stable than the second order extrapolation.

Triple point dynamics Two methods for updating the position of triple points have been tested. The dynamic contact angle method is based on heuristics and is fully geometrical. The zero shear stress model overcomes the kinematic paradox by allowing free slip at the triple point. In general we found the dynamic contact angle model to be more stable than the zero shear stress model. Selecting appropriate values of the tuning parameters seems to be easy.

Enforcing boundary conditions Two ways of enforcing the boundary conditions have been tested. In one case the values on the boundary are calculated explicitly for one node at a time, while in the other case a system of equations is generated for the simultaneous calculation of all unknowns along the boundary. We did not find any difference between the two methods with respect to stability or the time step allowed. So in general the explicit method is recommended, since it is much faster than the implicit one. It is mentioned, though, that the smoothing length for boundary nodes can be reduced using the implicit method.

Time integration method We have tested an explicit as well as an implicit time integration method. In general the explicit method was found to be faster than the implicit one. However, if the number of nodes is small and the spacial discretization is fine the implicit method becomes more attractive. For the problems presented in this work the explicit method was always superior with respect to computational cost.

Time step selection For the problems under consideration we have always used a fixed time step. However, by letting the step size vary there is potential of obtaining faster simulations. We have not investigated this subject in depth, so a recommendation for either fixed or variable time stepping cannot be given at present.

The work presented in this thesis is divided in two main categories, depending on whether the starting point is the Reynolds equation or the Navier–Stokes equations. In order to continue the present work a choice of strategy is needed. Either one can revert to using just the Reynolds equation, or one may continue working with the Navier–Stokes equations. It is not the aim of this text to perform

that selection. However, the following is concerned with the formulation using the Navier–Stokes equations.

The main difficulty using the Navier–Stokes equations is the induced computational cost. It has been realized that the feasible time step is of the order $\Delta t = 10^{-8}$ s, while the time required for a single revolution is 1 second. The suggested future work is therefore comprised of the following

1. Establish the size of the zone at the inlet where 2D effects need to be resolved by the Navier–Stokes equations
2. Establish whether or not the coupling with Reynolds equation in the present form is valid (the quasi–static assumption)
3. Introduce different discretization lengths in the x-direction and y-direction
4. Find out whether the inlet and outlet zones of the piston rings can be treated independently during one time step (decomposition)
5. Examine which of explicit or implicit time integration is the best choice, assuming the system can be decomposed
6. Use parallel computing techniques

A part from that is mentioned above a fully fledged piston ring simulation program would include computation of the lateral movement of the piston ring, deformation of the liner and the piston ring, frictional losses, wear rates, etc. These subjects are, however, outside the scope of this work.

Chapter 11

Conclusion

This work is concerned with the numerical simulation of piston ring lubrication. The main area of research is the description of the location of the inlet and outlet points on the piston ring. The work has been carried out from two different starting points, using either the Reynolds equation or the Navier–Stokes equations.

The first part of this thesis features a detailed derivation of the Reynolds equation. Special attention is given to the translation of Lagrangian velocity components into the Eulerian description needed for the Reynolds equation. The subject of cavitation is also treated. A new iteration procedure using the Newton–Raphson method has been proposed. An investigation of the inlet location has been carried out under the assumption that a steady-state condition exists. In this case it was found that back flow cannot take place as long as h_{in} is less than $3h_{\infty}$. Still assuming steady-state conditions a procedure for determining the inlet point, given the undisturbed oil film thickness and the running conditions of the piston ring, has been described. A concrete example using real life input data has been examined. The results show that in fact, the degree of lubrication for a piston ring can vary from fully flooded to starved conditions.

In the second part of this work the starting point has been the Navier–Stokes equations. A numerical model has been developed, which allows the simulation of the free surface outside the piston ring. A number of different formulations for the boundary conditions have been presented and examined. It was found that the Neumann boundary condition for density is more stable than the extrapolation methods. For updating the triple points the dynamic contact angle model is preferred over the zero shear stress model. Explicit as well as implicit time integration has been tested. Although the Jacobian is established fully analytically and simplified Newton–Raphson iterations are used the implicit time integration method was not competitive with respect to computational cost. The simulation program has been benchmarked using well-known test cases. The lid driven cavity problem shows that the spacial discretization has first order convergence, in accordance with theory. The fixed incline slider problem shows agreement of the pressure distribution between the simulation program and a reference solu-

tion obtained from the Reynolds equation. Finally, a rotating free surface and simulation of drop oscillations show that the free surface boundary condition has been implemented correctly. With respect to simulation of piston ring lubrication short-comings were encountered for the allowed time step. For a realistic problem the time step is of the magnitude 10^{-8} s, which is very little compared to 1 second, which is about the time needed for one revolution of the crank shaft in big Diesel engines. As an attempt to save computational work a combination of the Navier–Stokes and the Reynolds equation has been developed. This formulation still needs some validation, but looks promising and should be considered in the future work.

Appendix A

Sensitivity of free surface normal and curvature

The normal and curvature of a free surface are calculated as given in (9.64). Below we provide the sensitivity of the components of the normal and the curvature, with respect to the positions of the nodes on which they depend. Instead of giving the results as one long expression we utilize a number of dummy variables.

Common expressions quoted from (9.61)-(9.65)

$$
\begin{aligned}
x_{\mathrm{W}} &= x_{i-1} - x_i \\
x_{\mathrm{E}} &= x_{i+1} - x_i \\
y_{\mathrm{W}} &= y_{i-1} - y_i \\
y_{\mathrm{E}} &= y_{i+1} - y_i \\
\tau_{\mathrm{W}} &= -\sqrt{x_{\mathrm{W}}^2 + y_{\mathrm{W}}^2} \\
\tau_{\mathrm{E}} &= \sqrt{x_{\mathrm{E}}^2 + y_{\mathrm{E}}^2} \\
H &= \tau_{\mathrm{W}}\tau_{\mathrm{E}}^2 - \tau_{\mathrm{E}}\tau_{\mathrm{W}}^2 \\
x' &= \frac{x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2}{H} \\
x'' &= 2\frac{x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}}}{H} \\
y' &= \frac{y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2}{H} \\
y'' &= 2\frac{y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}}}{H} \\
L &= \sqrt{(x')^2 + (y')^2} \\
n_x &= -y'/L \\
n_y &= x'/L \\
\kappa &= \frac{x'y'' - y'x''}{L^3}
\end{aligned}
\tag{A.1}
$$

Sensitivities with respect to x_{i-1}

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}} &= \frac{x_{\mathrm{W}}}{\tau_{\mathrm{W}}} \\
\frac{\partial H}{\partial x_{i-1}} &= \frac{\tau_{\mathrm{W}}}{\partial x_{i-1}} \tau_{\mathrm{E}}^2 - 2\tau_{\mathrm{E}}\tau_{\mathrm{W}} \frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}} \\
\frac{\partial x'}{\partial x_{i-1}} &= \frac{(\tau_{\mathrm{E}}^2 - 2x_{\mathrm{E}}\tau_{\mathrm{W}} \frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}})H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial x_{i-1}}}{H^2} \\
\frac{\partial y'}{\partial x_{i-1}} &= \frac{(-2y_{\mathrm{E}}\tau_{\mathrm{W}} \frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}})H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{H}{\partial x_{i-1}}}{H^2} \\
\frac{\partial x''}{\partial x_{i-1}} &= 2\frac{(x_{\mathrm{E}} \frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}} - \tau_{\mathrm{E}})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_{i-1}}}{H^2} \\
\frac{\partial y''}{\partial x_{i-1}} &= 2\frac{(y_{\mathrm{E}} \frac{\partial \tau_{\mathrm{W}}}{\partial x_{i-1}})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_{i-1}}}{H^2} \\
\frac{\partial L}{\partial x_{i-1}} &= \frac{x' \frac{\partial x'}{\partial x_{i-1}} + y' \frac{\partial y}{\partial x_{i-1}}}{L} \\
\frac{\partial n_x^i}{\partial x_{i-1}} &= -\frac{\frac{\partial y'}{\partial x_{i-1}} L - y' \frac{\partial L}{\partial x_{i-1}}}{L^2} \\
\frac{\partial n_y^i}{\partial x_{i-1}} &= \frac{\frac{\partial x'}{\partial x_{i-1}} L - x' \frac{\partial L}{\partial x_{i-1}}}{L^2} \\
\frac{\partial \kappa}{\partial x_{i-1}} &= \frac{(\frac{\partial x'}{\partial x_{i-1}} y'' + x' \frac{\partial y''}{\partial x_{i-1}} - \frac{\partial y'}{\partial x_{i-1}} x'' - y' \frac{\partial x''}{\partial x_{i-1}})L^3 - (x'y'' - y'x'')3L^2 \frac{\partial L}{\partial x_{i-1}}}{L^6}
\end{aligned}
\tag{A.2}
$$

Sensitivities with respect to y_{i-1}

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}} &= \frac{y_{\mathrm{W}}}{\tau_{\mathrm{W}}} \\
\frac{\partial H}{\partial y_{i-1}} &= \frac{\tau_{\mathrm{W}}}{\partial y_{\mathrm{W}}}\tau_{\mathrm{E}}^2 - 2\tau_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}} \\
\frac{\partial x'}{\partial y_{i-1}} &= \frac{(-2x_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}})H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial y_{i-1}}}{H^2} \\
\frac{\partial y'}{\partial y_{i-1}} &= \frac{(\tau_{\mathrm{E}}^2 - 2y_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}})H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{H}{y_{i-1}}}{H^2} \\
\frac{\partial x''}{\partial y_{i-1}} &= 2\frac{(x_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_{i-1}}}{H^2} \\
\frac{\partial y''}{\partial y_{i-1}} &= 2\frac{(y_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_{i-1}} - \tau_{\mathrm{E}})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_{i-1}}}{H^2} \\
\frac{\partial L}{\partial y_{i-1}} &= \frac{x'\frac{\partial x'}{\partial y_{i-1}} + y'\frac{\partial y'}{\partial y_{i-1}}}{L} \\
\frac{\partial n_x^i}{\partial y_{i-1}} &= -\frac{\frac{\partial y'}{\partial y_{i-1}}L - y'\frac{\partial L}{\partial y_{i-1}}}{L^2} \\
\frac{\partial n_y^i}{\partial y_{i-1}} &= \frac{\frac{\partial x'}{\partial y_{i-1}}L - x'\frac{\partial L}{\partial y_{i-1}}}{L^2} \\
\frac{\partial \kappa}{\partial y_{i-1}} &= \frac{(\frac{\partial x'}{\partial y_{i-1}}y'' + x'\frac{\partial y''}{\partial y_{i-1}} - \frac{\partial y'}{\partial y_{i-1}}x'' - y'\frac{\partial x''}{\partial y_{i-1}})L^3 - (x'y'' - y'x'')3L^2\frac{\partial L}{\partial y_{i-1}}}{L^6}
\end{aligned}
\tag{A.3}
$$

Sensitivities with respect to x_i

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{W}}}{\partial x_i} &= -\frac{x_{\mathrm{W}}}{\tau_{\mathrm{W}}} \\
\frac{\partial \tau_{\mathrm{E}}}{\partial x_i} &= -\frac{x_{\mathrm{E}}}{\tau_{\mathrm{E}}} \\
\frac{\partial H}{\partial x_i} &= \frac{\tau_{\mathrm{W}}}{\partial x_i}\tau_{\mathrm{E}}^2 + 2\tau_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_i} - \frac{\partial \tau_{\mathrm{E}}}{\partial x_i}\tau_{\mathrm{W}}^2 - 2\tau_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial x_i} \\
\frac{\partial x'}{\partial x_i} &= \frac{(-\tau_{\mathrm{E}}^2 + 2x_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_i} + \tau_{\mathrm{W}}^2 - 2x_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial x_i})H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial x_i}}{H^2} \\
\frac{\partial y'}{\partial x_i} &= \frac{2(y_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_i} - y_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial x_i})H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial x_i}}{H^2} \\
\frac{\partial x''}{\partial x_i} &= 2\frac{(-\tau_{\mathrm{W}} + x_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial x_i} + \tau_{\mathrm{E}} - x_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_i})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_i}}{H^2} \\
\frac{\partial y''}{\partial x_i} &= 2\frac{(y_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial x_i} - y_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_i})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_i}}{H^2} \\
\frac{\partial L}{\partial x_i} &= \frac{x'\frac{\partial x'}{\partial x_i} + y'\frac{\partial y'}{\partial x_i}}{L} \\
\frac{\partial n_x^i}{\partial x_i} &= -\frac{\frac{\partial y'}{\partial x_i}L - y'\frac{\partial L}{\partial x_i}}{L^2} \\
\frac{\partial n_y^i}{\partial x_i} &= \frac{\frac{\partial x'}{\partial x_i}L - x'\frac{\partial L}{\partial x_i}}{L^2} \\
\frac{\partial \kappa}{\partial x_i} &= \frac{(\frac{\partial x'}{\partial x_i}y'' + x'\frac{\partial y''}{\partial x_i} - \frac{\partial y'}{\partial x_i}x'' - y'\frac{\partial x''}{\partial x_i})L^3 - (x'y'' - y'x'')3L^2\frac{\partial L}{\partial x_i}}{L^6}
\end{aligned}
\tag{A.4}
$$

Sensitivities with respect to y_i

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{W}}}{\partial y_i} &= -\frac{y_{\mathrm{W}}}{\tau_{\mathrm{W}}} \\
\frac{\partial \tau_{\mathrm{E}}}{\partial y_i} &= -\frac{y_{\mathrm{E}}}{\tau_{\mathrm{E}}} \\
\frac{\partial H}{\partial y_i} &= \frac{\tau_{\mathrm{W}}}{\partial y_i}\tau_{\mathrm{E}}^2 - 2\tau_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{\mathrm{P}}} - \frac{\tau_{\mathrm{E}}}{\partial y_{\mathrm{P}}}\tau_{\mathrm{W}}^2 - 2\tau_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_i} \\
\frac{\partial x'}{\partial y_i} &= \frac{2(x_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_i} - x_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\tau_{\mathrm{W}}}{\partial y_i})H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial y_i}}{H^2} \\
\frac{\partial y'}{\partial y_i} &= \frac{(-\tau_{\mathrm{E}}^2 + 2y_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_i} + \tau_{\mathrm{W}}^2 - 2y_{\mathrm{E}}\tau_{\mathrm{W}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_i})H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial y_i}}{H^2} \\
\frac{\partial x''}{\partial y_i} &= 2\frac{(x_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_i} - x_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_i})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_i}}{H^2} \\
\frac{\partial y''}{\partial y_i} &= 2\frac{(-\tau_{\mathrm{W}} + y_{\mathrm{E}}\frac{\partial \tau_{\mathrm{W}}}{\partial y_i} - \tau_{\mathrm{E}} - y_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_i})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_i}}{H^2} \\
\frac{\partial L}{\partial y_i} &= \frac{x'\frac{\partial x'}{\partial y_i} + y'\frac{\partial y'}{\partial y_i}}{L} \\
\frac{\partial n_x^i}{\partial y_i} &= -\frac{\frac{\partial y'}{\partial y_i}L - y'\frac{\partial L}{\partial y_i}}{L^2} \\
\frac{\partial n_y^i}{\partial y_i} &= \frac{\frac{\partial x'}{\partial y_i}L - x'\frac{\partial L}{\partial y_i}}{L^2} \\
\frac{\partial \kappa}{\partial y_i} &= \frac{(\frac{\partial x'}{\partial y_i}y'' + x'\frac{\partial y''}{\partial y_i} - \frac{\partial y'}{\partial y_i}x'' - y'\frac{\partial x''}{\partial y_i})L^3 - (x'y'' - y'x'')3L^2\frac{\partial L}{\partial y_i}}{L^6}
\end{aligned}
\tag{A.5}
$$

Sensitivities with respect to x_{i+1}

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}} &= \frac{x_{\mathrm{E}}}{\tau_{\mathrm{E}}} \\
\frac{\partial H}{\partial x_{i+1}} &= 2\tau_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}} - \frac{\tau_{\mathrm{E}}}{\partial x_{i+1}}\tau_{\mathrm{W}}^2 \\
\frac{\partial x'}{\partial x_{i+1}} &= \frac{(2x_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}} - \tau_{\mathrm{W}}^2)H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial x_{i+1}}}{H^2} \\
\frac{\partial y'}{\partial x_{i+1}} &= \frac{(2y_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}})H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial x_{i+1}}}{H^2} \\
\frac{\partial x''}{\partial x_{i+1}} &= 2\frac{(\tau_{\mathrm{W}} - x_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_{i+1}}}{H^2} \\
\frac{\partial y''}{\partial x_{\mathrm{E}}} &= 2\frac{(-y_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial x_{i+1}})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial x_{i+1}}}{H^2} \\
\frac{\partial L}{\partial x_{i+1}} &= \frac{x'\frac{\partial x'}{\partial x_{i+1}} + y'\frac{\partial y}{\partial x_{i+1}}}{L} \\
\frac{\partial n_x^i}{\partial x_{i+1}} &= -\frac{\frac{\partial y'}{\partial x_{i+1}}L - y'\frac{\partial L}{\partial x_{i+1}}}{L^2} \\
\frac{\partial n_y^i}{\partial x_{i+1}} &= \frac{\frac{\partial x'}{\partial x_{i+1}}L - x'\frac{\partial L}{\partial x_{i+1}}}{L^2} \\
\frac{\partial \kappa}{\partial x_{i+1}} &= \frac{(\frac{\partial x'}{\partial x_{i+1}}y'' + x'\frac{\partial y''}{\partial x_{i+1}} - \frac{\partial y'}{\partial x_{i+1}}x'' - y'\frac{\partial x''}{\partial x_{i+1}})L^3 - (x'y'' - y'x'')3L^2\frac{\partial L}{\partial x_{i+1}}}{L^6}
\end{aligned}
\tag{A.6}
$$

Sensitivities with respect to y_{i+1}

$$
\begin{aligned}
\frac{\partial \tau_{\mathrm{E}}}{\partial y_{i+1}} &= \frac{y_{\mathrm{E}}}{\tau_{\mathrm{E}}} \\
\frac{\partial H}{\partial y_{i+1}} &= 2\tau_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{\mathrm{E}}} - \frac{\tau_{\mathrm{E}}}{\partial y_{\mathrm{E}}}\tau_{\mathrm{W}}^2 \\
\frac{\partial x'}{\partial y_{i+1}} &= \frac{(2x_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{i+1}})H - (x_{\mathrm{W}}\tau_{\mathrm{E}}^2 - x_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial y_{i+1}}}{H^2} \\
\frac{\partial y'}{\partial y_{i+1}} &= \frac{(2y_{\mathrm{W}}\tau_{\mathrm{E}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{i+1}} - \tau_{\mathrm{W}}^2)H - (y_{\mathrm{W}}\tau_{\mathrm{E}}^2 - y_{\mathrm{E}}\tau_{\mathrm{W}}^2)\frac{\partial H}{\partial y_{i+1}}}{H^2} \\
\frac{\partial x''}{\partial y_{i+1}} &= 2\frac{(-x_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{i+1}})H - (x_{\mathrm{E}}\tau_{\mathrm{W}} - x_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_{i+1}}}{H^2} \\
\frac{\partial y''}{\partial y_{i+1}} &= 2\frac{(\tau_{\mathrm{W}} - y_{\mathrm{W}}\frac{\partial \tau_{\mathrm{E}}}{\partial y_{i+1}})H - (y_{\mathrm{E}}\tau_{\mathrm{W}} - y_{\mathrm{W}}\tau_{\mathrm{E}})\frac{\partial H}{\partial y_{i+1}}}{H^2} \\
\frac{\partial L}{\partial y_{i+1}} &= \frac{x'\frac{\partial x'}{\partial y_{i+1}} + y'\frac{\partial y'}{\partial y_{i+1}}}{L} \\
\frac{\partial n_x^i}{\partial y_{i+1}} &= -\frac{\frac{\partial y'}{\partial y_{i+1}}L - y'\frac{\partial L}{\partial y_{i+1}}}{L^2} \\
\frac{\partial n_y^i}{\partial y_{i+1}} &= \frac{\frac{\partial x'}{\partial y_{i+1}}L - x'\frac{\partial L}{\partial y_{i+1}}}{L^2} \\
\frac{\partial \kappa}{\partial y_{i+1}} &= \frac{(\frac{\partial x'}{\partial y_{i+1}}y'' + x'\frac{\partial y''}{\partial y_{i+1}} - \frac{\partial y'}{\partial y_{i+1}}x'' - y'\frac{\partial x''}{\partial y_{i+1}})L^3 - (x'y'' - y'x'')3L^2\frac{\partial L}{\partial y_{i+1}}}{L^6}
\end{aligned}
\tag{A.7}
$$

Appendix B

Comparison of Jacobian matrices

In this section we show plots of the Jacobian matrix, associated with the rotating disc problem from section 10.3. Because the number of variables is big, we split the state vector into groups of variables. We when plot the components of the analytical Jacobian and also the difference between the analytical Jacobian and the same matrix generated by a finite difference approximation. The following plots are provided

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right)$ figure B.1

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}y_i}{\mathrm{d}t}\right)$ figure B.2

$\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.3

$\frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.4

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.5

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.6

$\frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.7

$\frac{\partial}{\partial x_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.8

$\frac{\partial}{\partial y_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)$ figure B.9

$\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.10

$\frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.11

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.12

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.13

$\frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.14

$\frac{\partial}{\partial x_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.15

$\frac{\partial}{\partial y_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)$ figure B.16

$\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.17

$\frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.18

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.19

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.20

$\frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.21

$\frac{\partial}{\partial x_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.22

$\frac{\partial}{\partial y_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}\rho_i}{\mathrm{d}t}\right)$ figure B.23

$\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.24

$\frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.25

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.26

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.27

$\frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.28

$\frac{\partial}{\partial x_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.29

$\frac{\partial}{\partial y_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}x_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.30

$\frac{\partial}{\partial x_j}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.31

$\frac{\partial}{\partial y_j}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.32

$\frac{\partial}{\partial u_j}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.33

$\frac{\partial}{\partial v_j}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.34

$\frac{\partial}{\partial \rho_j}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.35

$\frac{\partial}{\partial x_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.36

$\frac{\partial}{\partial y_j^{\mathrm{fs}}}\left(\frac{\mathrm{d}y_i^{\mathrm{fs}}}{\mathrm{d}t}\right)$ figure B.37

Note that some plots of the gradients of the time derivatives of x_i and y_i are omitted – this has been done because the gradients are identically zero.

Superscript $^{\mathrm{fs}}$ indicates that a node/function belongs to a *free surface*. Variables/functions without a superscript are associated with *fluid* nodes.

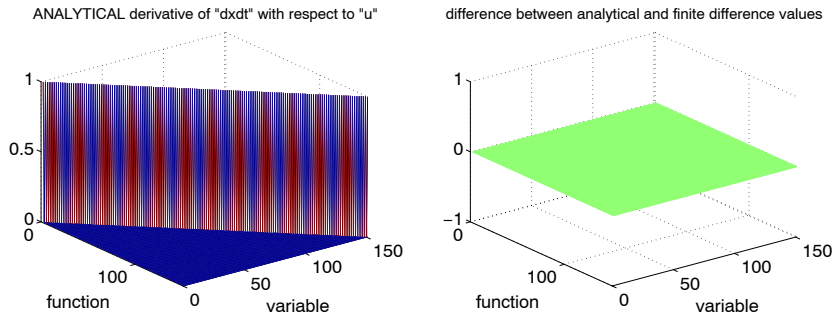

Figure B.1: $\partial(\mathrm{d}x_i/\mathrm{d}t)/\partial u_j$

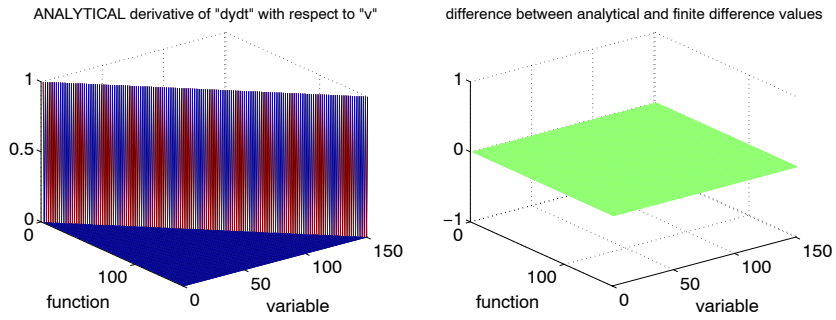

Figure B.2: $\partial(\mathrm{d}y_i/\mathrm{d}t)/\partial v_j$

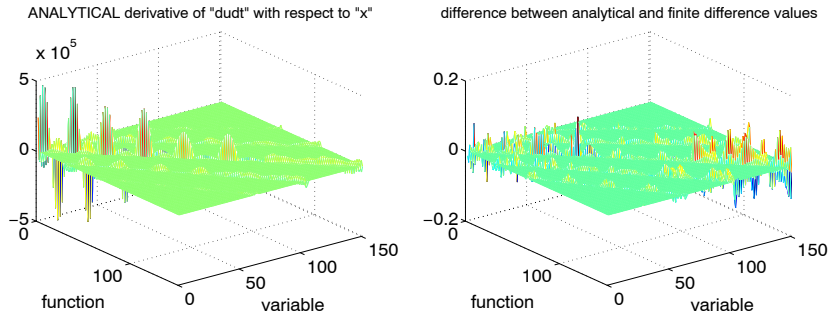

Figure B.3: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial x_j$

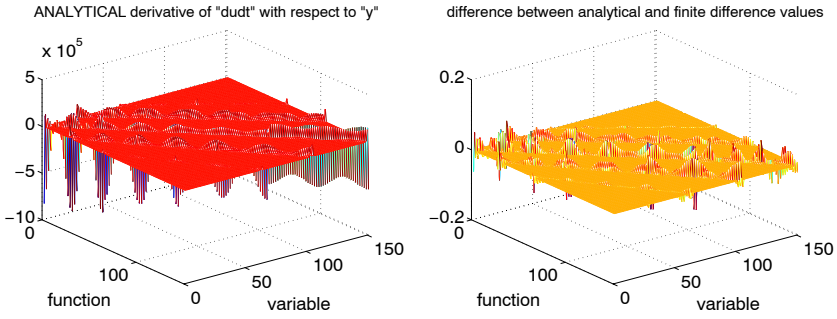

Figure B.4: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial y_j$

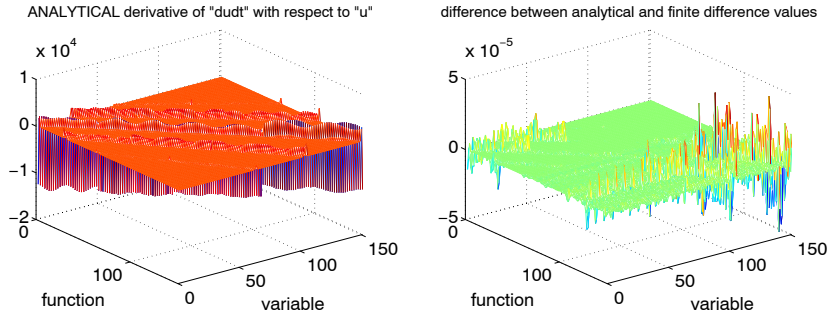

Figure B.5: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial u_j$

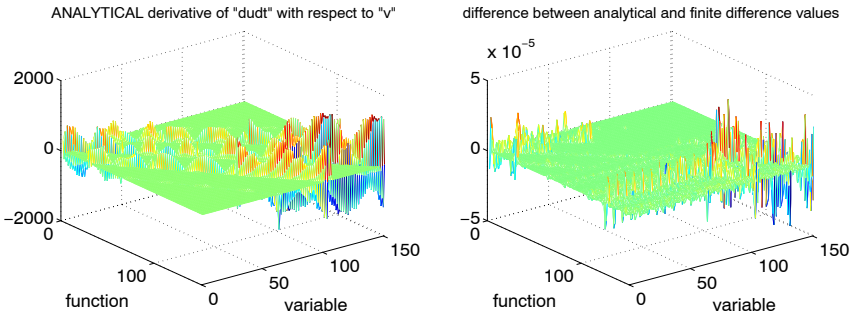

Figure B.6: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial v_j$

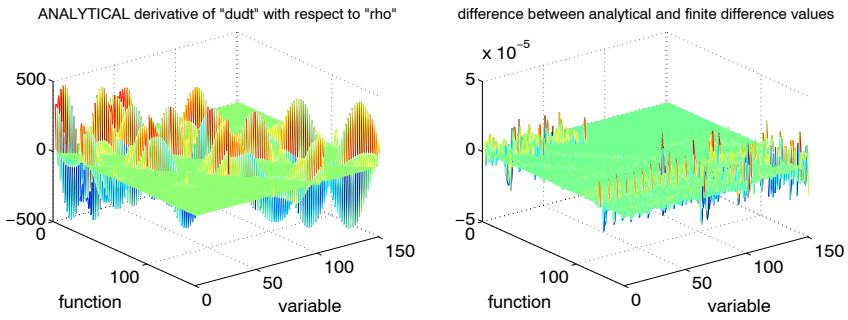

Figure B.7: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial \rho_j$

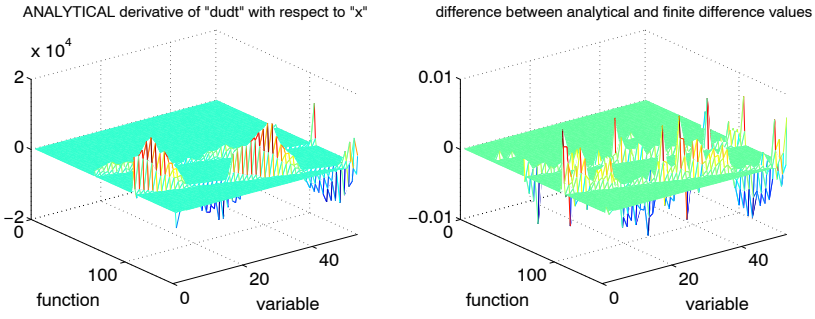

Figure B.8: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial x_j^{\mathrm{fs}}$

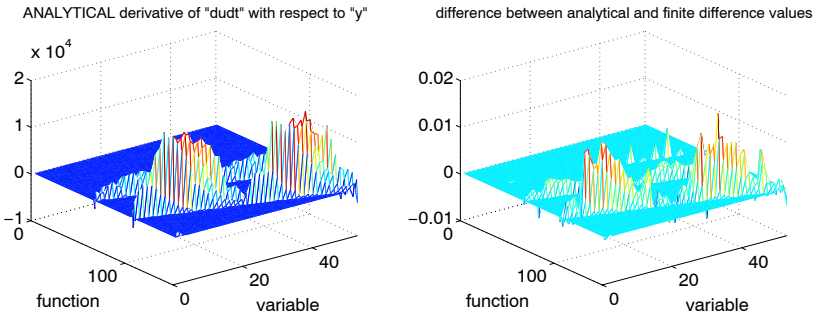

Figure B.9: $\partial(\mathrm{d}u_i/\mathrm{d}t)/\partial y_j^{\mathrm{fs}}$

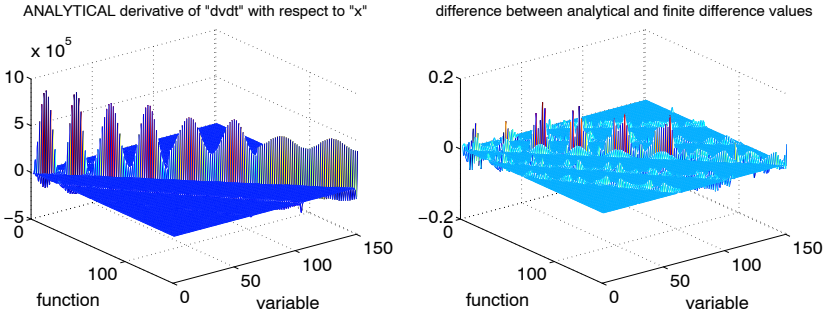

Figure B.10: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial x_j$

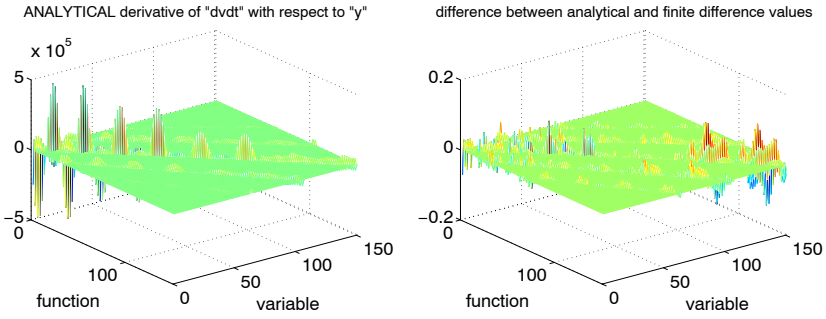

Figure B.11: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial y_j$

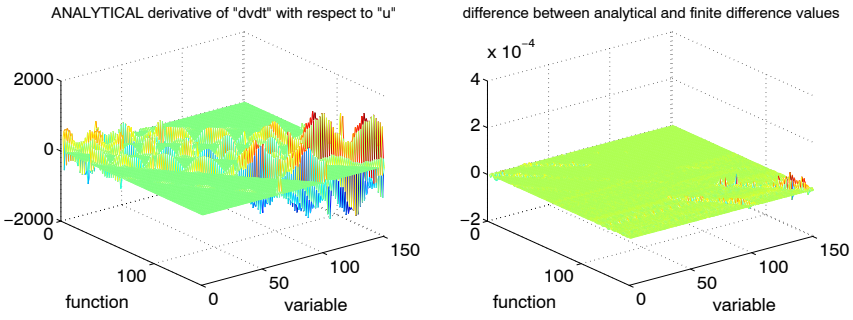

Figure B.12: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial u_j$

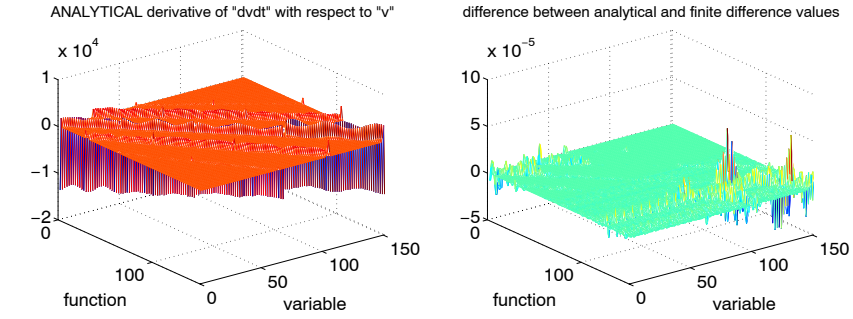

Figure B.13: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial v_j$

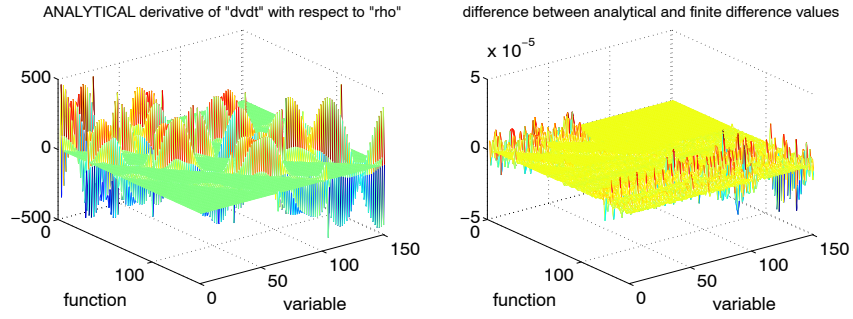

Figure B.14: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial \rho_j$

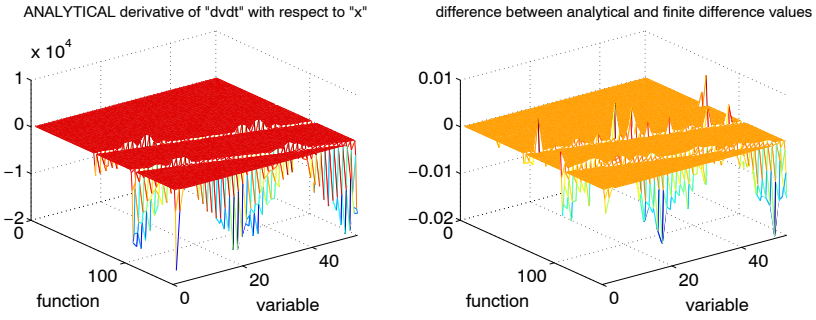

Figure B.15: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial x_j^{\mathrm{fs}}$

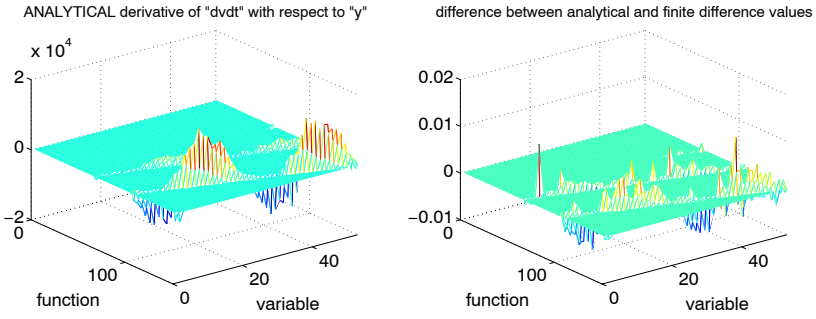

Figure B.16: $\partial(\mathrm{d}v_i/\mathrm{d}t)/\partial y_j^{\mathrm{fs}}$

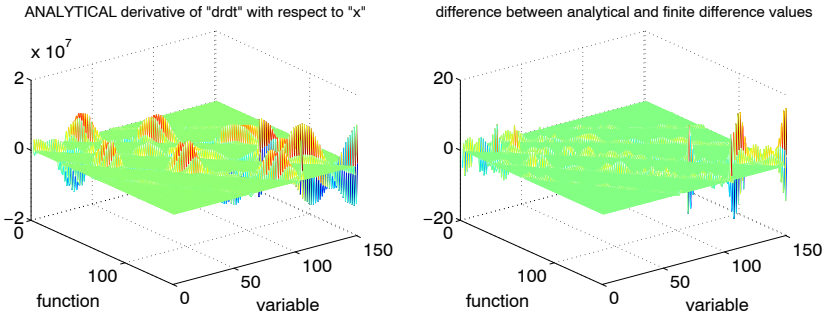

Figure B.17: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial x_j$

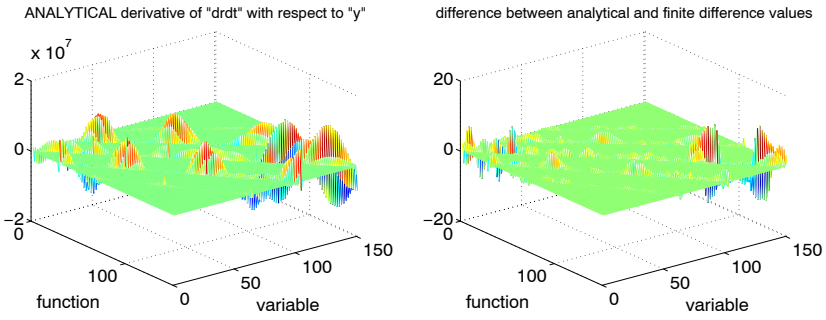

Figure B.18: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial y_j$

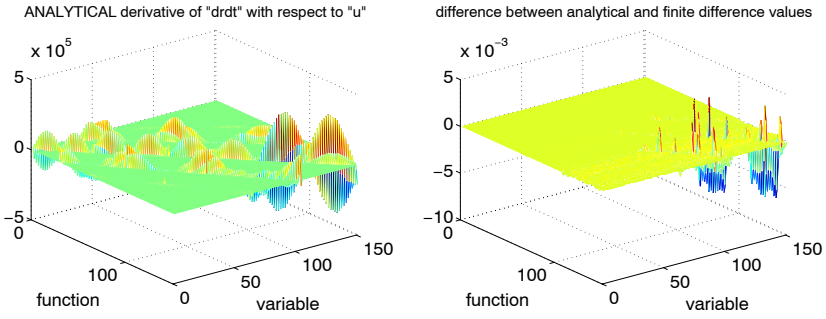

Figure B.19: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial u_j$

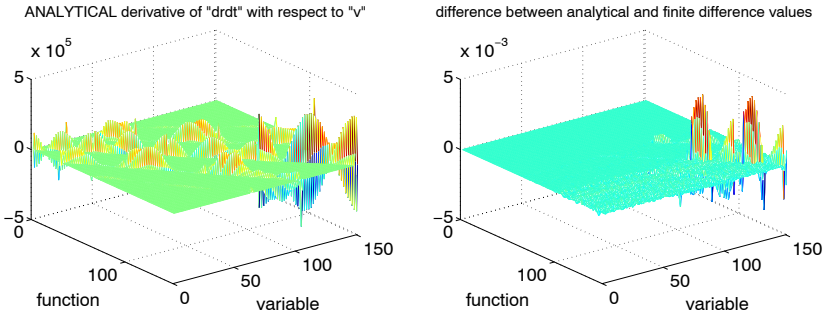

Figure B.20: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial v_j$

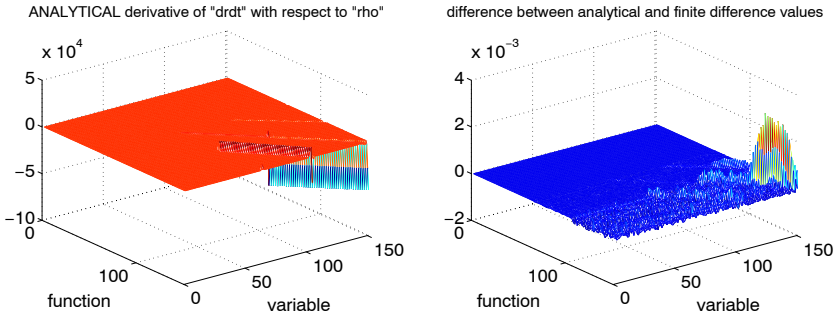

Figure B.21: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial\rho_j$

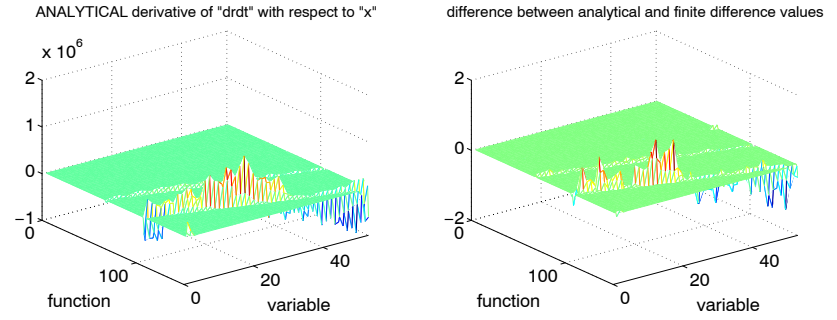

Figure B.22: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial x_j^{\mathrm{fs}}$

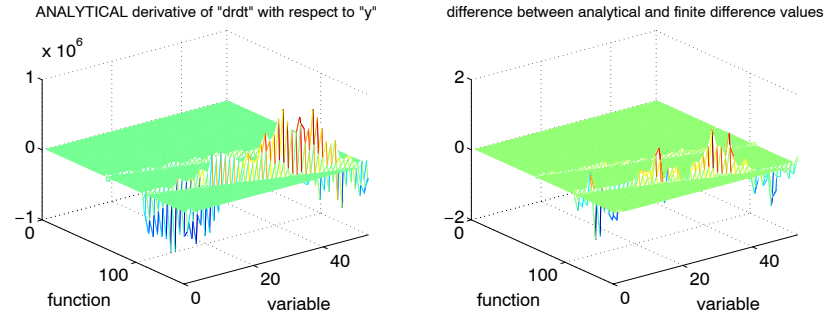

Figure B.23: $\partial(\mathrm{d}\rho_i/\mathrm{d}t)/\partial y_j^{\mathrm{fs}}$

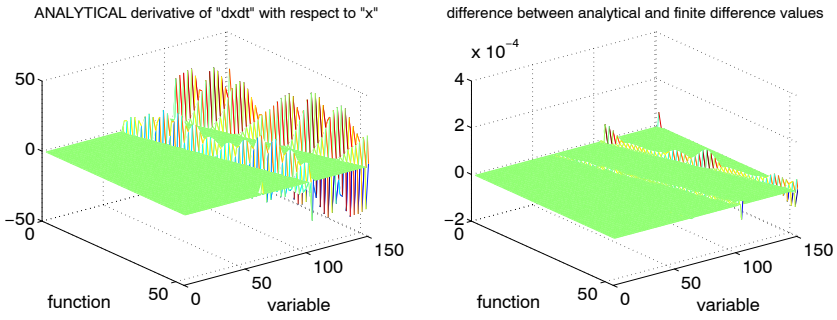

Figure B.24: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial x_j$

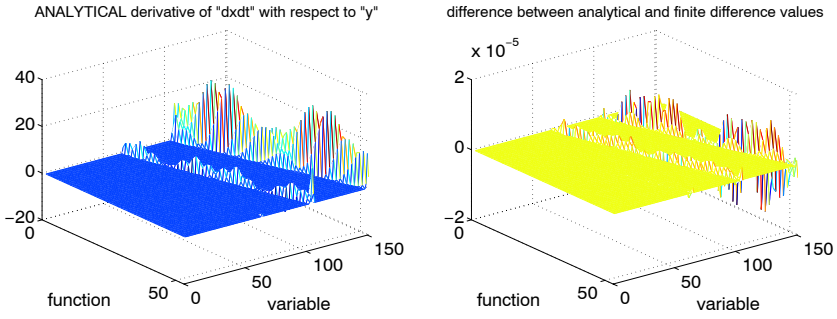

Figure B.25: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial y_j$

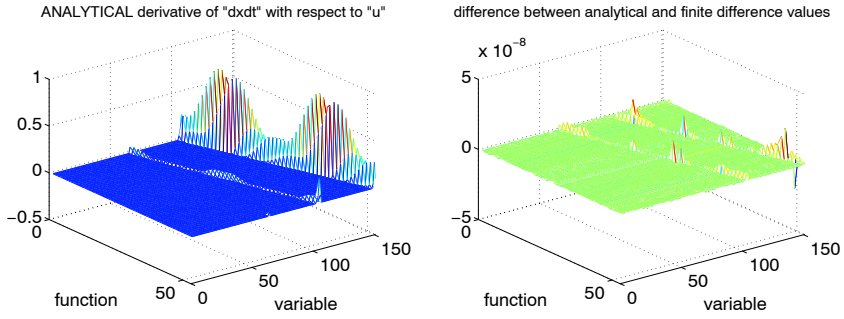

Figure B.26: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial u_j$

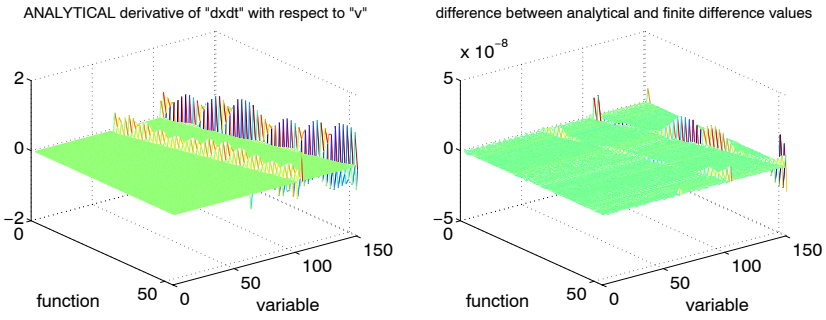

Figure B.27: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial v_j$

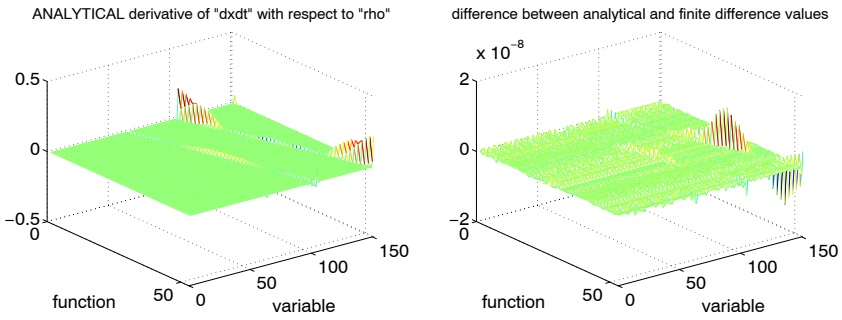

Figure B.28: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial \rho_j$

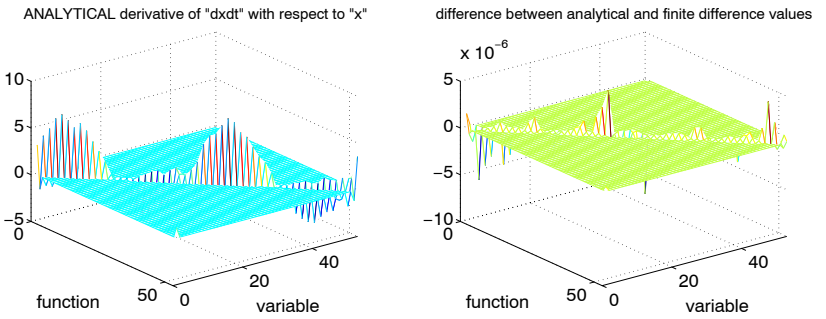

Figure B.29: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial x_j^{\mathrm{fs}}$

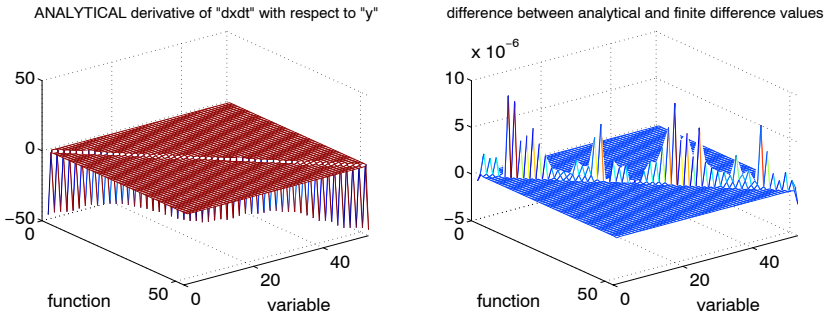

Figure B.30: $\partial(\mathrm{d}x_i^{\mathrm{fs}}/\mathrm{d}t)/\partial y_j^{\mathrm{fs}}$

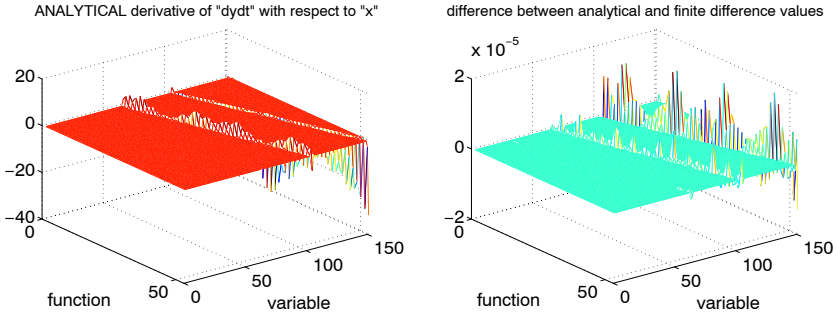

Figure B.31: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial x_j$

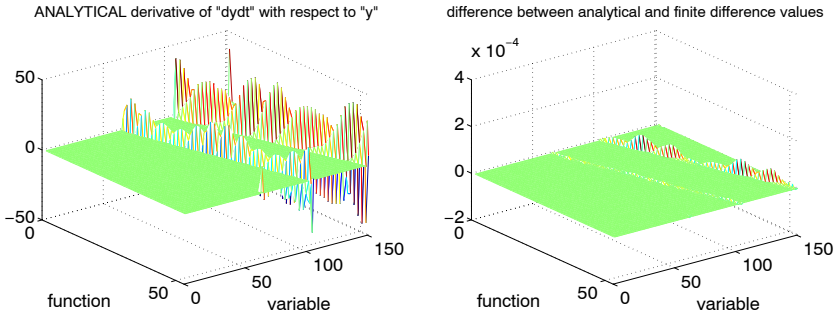

Figure B.32: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial y_j$

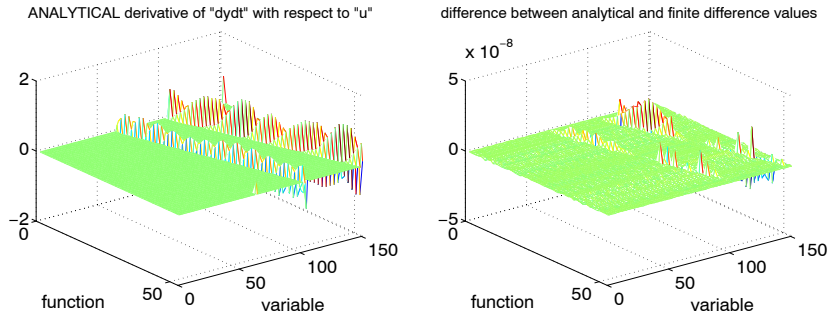

Figure B.33: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial u_j$

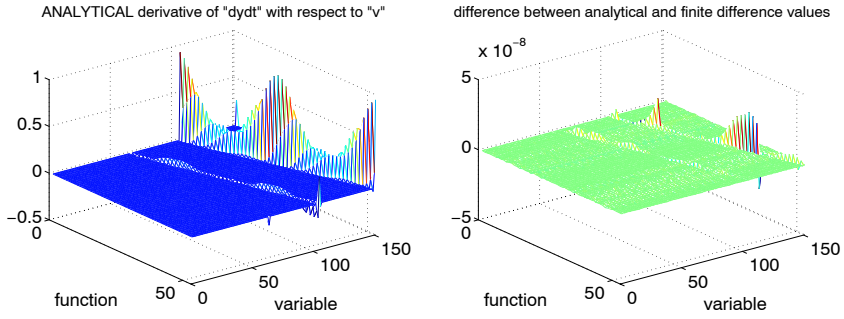

Figure B.34: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial v_j$

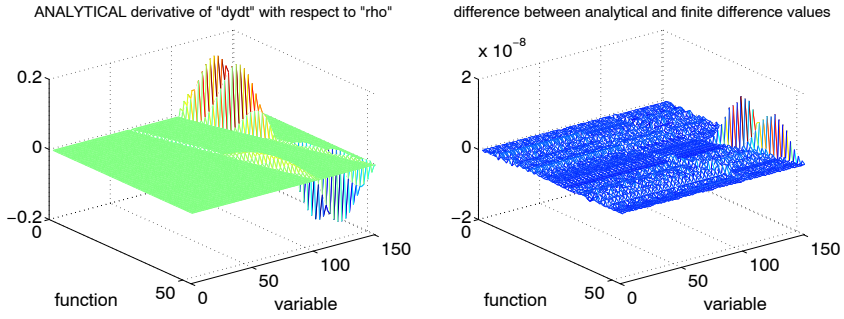

Figure B.35: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial \rho_j$

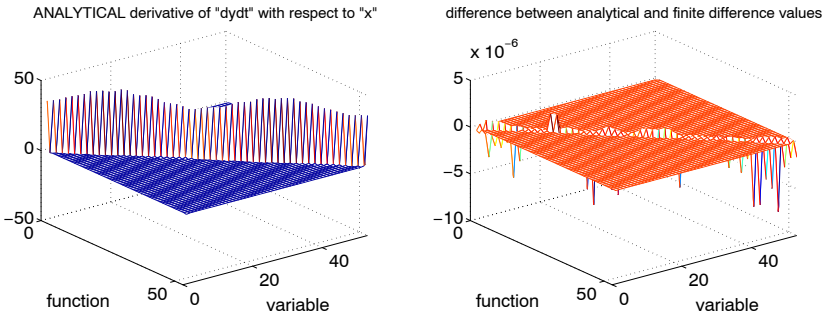

Figure B.36: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial x_j^{\mathrm{fs}}$

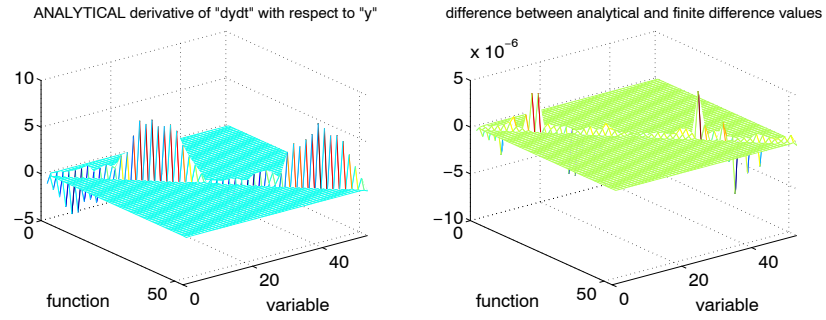

Figure B.37: $\partial(\mathrm{d}y_i^{\mathrm{fs}}/\mathrm{d}t)/\partial y_j^{\mathrm{fs}}$

Appendix C

On the finite difference approximation of $\mathrm{d}h/\mathrm{d}x$

During the defense of this thesis attention was drawn to the finite difference approximation of $\mathrm{d}h/\mathrm{d}x$. A little bit of analysis will reveal the truncation error. First we consider the method used in this thesis

$$\frac{\mathrm{d}(h^3)}{\mathrm{d}x}|_{x=x_i} \approx \frac{(h_{i+1})^3 - (h_{i-1})^3}{2\Delta x} \tag{C.1}$$

Substituting first order Taylor expansions of h_{i+1} and h_{i-1} gives

$$\begin{aligned}\frac{(h_{i+1})^3 - (h_{i-1})^3}{2\Delta x} &= \frac{(h_i + h_i'\Delta x)^3 - (h_i - h_i'\Delta x)^3}{2\Delta x} \\ &= \frac{6(h_i)^2 h_i'\Delta x + 2(h_i')^3(\Delta x)^3}{2\Delta x} \\ &= 3(h_i)^2 h_i' + (h_i')^3(\Delta x)^2\end{aligned}$$

From this it is seen that the truncation error is proportional to $(\mathrm{d}h/\mathrm{d}x)^3$. This situation is not desirable, especially if $\mathrm{d}h/\mathrm{d}x$ is large.

If we instead manipulate in the following way

$$\frac{\mathrm{d}(h^3)}{\mathrm{d}x}|_{x=x_i} = 3(h_i)^2\frac{\mathrm{d}h}{\mathrm{d}x}|_{x=x_i} \approx 3(h_i)^2\frac{h_{i+1} - h_{i-1}}{2\Delta x} \tag{C.2}$$

we get using a third order Taylor expansion

$$\begin{aligned}
3(h_i)^2\frac{h_{i+1}-h_{i-1}}{2\Delta x} &= 3(h_i)^2\frac{\left(h_i + h_i'\Delta x + h_i''\dfrac{(\Delta x)^2}{2} + h_i'''\dfrac{(\Delta x)^3}{6}\right)}{2\Delta x} \\
&\quad - 3(h_i)^2\frac{\left(h_i - h_i'\Delta x + h_i''\dfrac{(\Delta x)^2}{2} - h_i'''\dfrac{(\Delta x)^3}{6}\right)}{2\Delta x} \\
&= 3(h_i)^2\frac{2h_i'\Delta x + h_i'''\dfrac{(\Delta x)^3}{3}}{2\Delta x} \\
&= 3(h_i)^2h_i' + 3(h_i)^2h_i'''\frac{(\Delta x)^2}{6}
\end{aligned}$$

In this case we obtain the familiar truncation error for second order finite difference approximations – that is, proportionality with $\mathrm{d}^3h/\mathrm{d}x^3$. Thus, we may conclude that (C.2) has better properties than (C.1), so one might consider to replace the current discretization of the Reynolds equation with expressions of the same kind as in (C.2).

Bibliography

[1] Mohsen Esfahanian, Bernard J. Hamrock, Abdallah A. Elsharkawy (1998): On the Hydrodynamic Lubrication Analysis of Piston Rings, Lubrication Science

[2] D. Dowson, P. N. Economou, B. L. Ruddy, P. J. Strachan, A. J. S. Baker (1979): Piston Ring Lubrication – Part II Theoretical Analysis of a Single Ring and a Complete Ring Pack, Energy Conservation Through Fluid Film Lubrication Technology: Frontiers in Research and Design, Proceedings of the ASME Winter Annual Meeting, pp. 23-52

[3] Dong-Chul Han, Jae-Seon Lee (1998): Analysis of the piston ring lubrication with a new boundary condition , Tribology International, Vol 31, No. 12, pp. 753-760

[4] R. J. Gamble, M. Priest, C. M. Taylor (2003): Detailed analysis of oil transport in the piston assembly of a gasoline engine, Tribology Letters, Vol. 14, No. 2, pp. 147-156

[5] C. W. Hirt, B. D. Nichols (1981): Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries, Journal of Computational Physics, 39, 201-225

[6] J. H. Ferziger, M. Peric (2002): Computational Methods for Fluid Dynamics, 3rd Edition, Springer

[7] T. J. Chung (2002): Computational Fluid Dynamics, Cambridge University Press

[8] —, p. 60

[9] Sin-Chung Chang (1995): The Method of Space–Time Conservation Element and Solution Element — A New Approach for Solving the Navier–Stokes and Euler Equations, Journal of Computational Physics, 119, 295-324

[10] Songdong Shao, Edmond Y. M. Lo (2003): Incompressible SPH method for simulating Newtonian and non-Newtonian flows with a free surface, Advances in Water Ressources 26, 787-800

[11] Andrea Colagrossi (2004): A Meshless Lagrangian Method for Free-Surface and Interface Flows with Fragmentation, PhD thesis, University of Rome, p. 26

[12] T. Belytschko, Y. Krongauz, D. Organ, M. Flemming, P. Krysl (1996): Meshless Methods: An Overview and Recent Developments

[13] G. R. Liu, M. B. Liu (2003): Smoothed Particle Hydrodynamics, a meshfree particle method, World Scientific

[14] —, pp. 105-124

[15] G. R. Liu (2003): Mesh Free Methods, Moving beyond the Finite Element Method, CRC Press

[16] —, pp. 70-74

[17] —, pp. 77-78

[18] M. Priest, D. Dowson, C. M. Taylor (2000): Theoretical modelling of cavitation in piston ring lubrication, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, Vol. 214, Issue. 3

[19] Parviz E. Nikravesh (1988): Computer-Aided Analysis of Mechanical Systems, Prentice Hall

[20] Ahmed A. Shabana (1998): Dynamics of Multibody Systems, Cambridge University Press

[21] Lennart Råde, Bertil Westergren (1990): BETA, Mathematics Handbook, Second Edition, Chartwell-Bratt Ltd, p. 348

[22] Bernard J. Hamrock (1994): Fundamentals of Fluid Film Lubrication, 3rd edition, McGraw-Hill, Inc, p. 175

[23] —, pp. 141-147

[24] —, pp. 229-230

[25] —, p. 71

[26] Ramsey Gohar (2001): Elastohydrodynamics, Second Edition, Imperial College Press, pp. 64-66

[27] Dan S. Henningson, Martin Berggren (2005): Lecture notes on Computational Fluid Dynamics, KTH

[28] J. W. Lund: Notater til forelæsninger i smøringsmekanik

[29] Steven H. Strogatz (1994): Nonlinear Dynamics and Chaos, Perseus Books, Cambridge, Massachusetts

[30] Matlab,
www.mathworks.com

[31] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery (2002): Numerical Recipes in C++, The Art of Scientific Computing, 2nd Edition, Cambridge University Press, pp. 401-406

[32] Microsoft Visual Studio C++,
http://msdn2.microsoft.com/en-us/visualc/default.aspx

[33] Unsymmetric MultiFrontal method,
http://www.cise.ufl.edu/research/sparse/umfpack

[34] GNU + Cygnus + Windows = cygwin, www.cygwin.com

[35] Automatically Tuned Linear AlgebraSoftware,
http://math-atlas.sourceforge.net

[36] OpenSource alternative to the OpenGL Utility Toolkit,
http://freeglut.sourceforge.net/

[37] Karl Brenkert, Jr. (1960): Elementary Theoretical Fluid Mechanics, John Wiley & Sons, Inc, Appendix A

[38] J. J. Monaghan, A. Kos, N. Issa (2003): Fluid Motion Generated by Impact, Journal of Waterway, Port, Costal and Ocean Engineering

[39] Graham B. Wallis (1969): One-dimensional Two-phase Flow, McGraw-Hill, p. 143

[40] David P- Schimdt, Christopher J. Rutland, M. L. Corradini (1991): Fully compressible, two-dimensional model of small, high–speed, cavitating nozzles, Atomization and Sprays, Journal of the International Institutions for Liquid Atomization and Spray Systems, Vol. 9, Issue.3, pp. 255-276

[41] David P. Schmidt, C. J. Rutland, M. L. Corradini, P. Roosen, O. Genge (1999): Cavitation in Two-Dimensional Asymmetric Nozzles, SAE Technical Paper Series, Reprint From: Technology for Diesel Fuel Injection and Sprays (SP-1415)

[42] Ciro Pascarella, Vito Salvatore, Alessandro Ciucci (2003): Effects of Speed of Sound Variation on Unsteady Cavitating Flows by using a Barotropic Model, Fifth International Symposium on Cavitation (CAV2003)

[43] Thomas A. Bear, Richard A. Cairncross, P. Randall Schunk, Rehka R. Rao, Phillip A. Sackinger (2000): A finite element method for free surface flows of incompressible fluids in three dimensions. Part II. Dynamic wetting lines, Int. J. Numer. Meth. Fluids 2000; 33; 405-427

[44] I. Robertson, S. J. Sherwin, J. M. R. Graham (2004): Comparison of wall boundary conditions for numerical viscous free surface flow simulation, Journal of Fluids and Structures 19 (2004) 525-542

[45] C. Armando Duarte (1995): A Review of Some Meshless Methods to Solve Partial Differential Equations, TICAM Report 95-06

[46] Shaofan Li, Wing Kam Liu (2002): Meshfree and particle methods and their applications, Appl Mech Rev vol 55, no 1

[47] Arieh Iserles (2003): A first course in the Numerical Analysis of Differential Equations, Cambridge University Press, p. 40

[48] Roger Alexander (2003): Design and implementation of DIRK integrators for stiff systems, Applied Numerical Mathematics 46 (2003) 1-17

[49] Lord Rayleigh (1879): On the Capillary Phenomena of Jets, Proceedings of the Royal Society of London, Vol. 29. pp. 71-97